WRDC-TM-90-315

VAX MERCURY User's Manual

Jed E. Marquart
WRDC/FIMM
Wright-Patterson AFB, OH 45433-6553

April 1990

Aerodynamic Methods Group
Aerodynamics and Airframe Branch
Aeromechanics Division

FLIGHT DYNAMICS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433

Foreword

This report describes the modifications to, and the operation of, the Euler flow solver code, MERCURY, and the auxiliary code, SETUP, as implemented for use on the Digital Equipment VAX series of computers. The original version of MERCURY and SETUP were coded by William Z. Strang of WRDC/FIMM. MERCURY was optimized, in terms of input/output (I/O) processing, data storage, and vector operations, for use on the CRAY series of supercomputers.

Due to interest from potential users of the code who did not have access to a CRAY for various reasons, a version of MERCURY was developed for use on an the VAX computers. The primary difference, from the point of view of the user (in addition to the processing speed), between the CRAY and VAX versions of MERCURY is in the method of invoking the code for execution.

The auxiliary code, SETUP, was rehosted onto the VAX, and modified to create the data files required by the VAX version of MERCURY. This modified version of SETUP is called SETUPV.

This report, then, describes the method of developing the grid, creating the necessary files for MERCURY, and executing the code.

Thanks are in order to William Z. Strang, for his patient assistance in explaining the AQIO (CRAY I/O) routines.

The work reported herein was performed during the period 1 December 1989 to 30 April 1990.

This work has been reviewed and is approved.

Jed E. Marquart

Jed E. Marquart
Aeronautical Engineer
Aerodynamic Methods Group

Don W. Kinsey

Dr. Don Kinsey
Group Leader
Aerodynamic Methods Group

Dennis Sedlock

Dennis Sedlock
Chief, Aerodynamics and Airframe Branch

Section I

Introduction

This report describes the use and operation of the MERCURY flow solver code on the Digital Equipment VAX series of computers. MERCURY is a multiple grid block flow solver which solves the Euler set of equations (inviscid flow).

The MERCURY code was developed and written by William Z. Strang of WRDC/FIMM for use on the CRAY supercomputer (reference Appendix D). As such, it was optimized to take full advantage of CRAY-specific I/O routines which greatly reduce the I/O time required. Since a large percentage of execution time is spent in the I/O process when using a multiblock code such as this, the optimization of the I/O process is essential to achieving acceptable turnaround times for the solution.

In addition to the CRAY-specific I/O routines, MERCURY was developed using one-dimensional arrays to reduce memory overhead and improve vectorization.

While the optimization of the I/O routines for operation on the CRAY makes MERCURY an extremely fast code, it also makes the code much less portable. Other machines do not permit the type of I/O which makes the CRAY so desirable. Thus, when it is desired to convert the code to run on another host machine, the I/O routines must be changed to work on the new host, while still maintaining the correct data structure and logical order.

The changes made to the MERCURY I/O routines are outlined in the following section.

Section II

CRAY to VAX Conversion of MERCURY

I/O Changes

The Digital Equipment VAX series of computers is a very popular and well-supported line of computational machines. Since they are established in industry, many offices have them, either in the smaller, "personal" versions (i.e. the MicroVAX) or in larger versions. In addition to their cost effectiveness and general popularity, they are well suited to placing in the types of environments required for secure processing.

One capability that the VAX does not possess, however, is that of extremely fast I/O such as the type which exists on the CRAY supercomputer, called Asynchronous Queued I/O (AQIO). The VAX, through FORTRAN, may use only the standard I/O routines available to FORTRAN. Thus, all AQIO routines in MERCURY needed to be changed to standard I/O routines for the VAX version.

These VAX FORTRAN routines, however, still need to input and output the data in the order that the MERCURY code expects it, in order to avoid major recoding and restructuring of MERCURY. Thus, a major portion of the translation of MERCURY from the CRAY version to the VAX version involved the work of restructuring the I/O calls to produce files which would be usable by MERCURY, but that were created using standard FORTRAN I/O routines.

The type of I/O used is called "direct access I/O", and is one of the standard types of data I/O used on machines such as the VAX. The files

produced by direct access I/O may contain any number of user-specified-length records, each of which is directly accessible (hence the name) by reference to its record number.

Section III

Running MERCURY on the VAX

Creating the Grid

As with any CFD flow solver, the first step in the solution process is to produce a grid which discretizes the solid surface and flow field geometries. This grid may be created through any means desired, provided that the output file matches the configuration requirements listed in the MERCURY User's Manual (Appendix D).

Some sort of grid evaluation tool should be run on the grid to be sure that the grid does not contain lefthanded portions, negative cell volumes, etc.

The grid may be of either binary or ASCII format, but be warned that an ASCII file is much larger than a binary file which contains the same data. In addition, reading the grid data from an ASCII file will take considerably longer than reading it from a binary file, although this only happens once at the beginning of each run. The flow solver data contains a switch to tell the flow solver whether the grid is in ASCII or binary.

Once the grid has been created, the flow solver job file may be filled in to match the desired conditions and run parameters.

Creating the Flow Solver Job File

The flow solver job file (example in Appendix B) contains the VMS commands necessary to compile and/or execute a version of MERCURY for a particular flow configuration, as well as the data necessary to

describe the flow conditions, the dimensions of the grid, and the parameters for the particular computer run.

The layout of the flow solver job file on the VAX is somewhat similar to that on the CRAY, in that the flow data is embedded in the same file as the commands which invoke the code.

The first three lines of the flow solver data (which is embedded in the flow solver job file code) contain the full VAX pathnames of the output scratch file, the input scratch file, and the restart file, respectively. Note that full pathnames must be used. The MERCURY code reads these file names, and assigns them to the appropriate I/O device numbers.

The input and output scratch files are the files which MERCURY uses to read and write the "temporary" flow variable data during execution. These files exist only during the code execution, and are removed by the VAX operating system after the completion of the code run. The only caution here is to be sure that the path specified for these files contains enough empty space for the files. Should the files overrun the empty space, the code will crash, since no error checking is performed for this circumstance. See Appendix D for details on the memory and storage requirements for MERCURY runs.

The restart file is the file into which all grid and flow data is written at the end of a run. This file is very important, since it is the basis for the start of the next run. If this file should be destroyed, all work done by the previous run(s) will go with it, and it will be time to start all over. Be sure that the path specified for this file has plenty of empty space.

The rest of the data contained in the flow solver data file is well documented in the MERCURY User's Manual. The only deviation on the VAX version from the CRAY version is that the VAX version does not permit "I/O DEVICE" selection. On the CRAY version of MERCURY, this option told the code whether the user wanted to use the standard hard disk or the solid-state storage device (SSD). Since the VAX does not support an SSD, this option was removed from the data input.

Creating the Connectivity and Plot Files

For the CRAY version of MERCURY, two data files are required which contain the data relating the connections between the grid blocks and the desired plot data. Each of these two files contains two portions, a "parameter" portion, which basically contains the necessary dimensions, and a data portion, which contains the actual data. These data files are created using the auxiliary program SETUP.

The CRAY has a means of splitting out the required data from the two data files to use as the four "include" files in the code. The VAX does not support the same type of structure. Thus, the four "include" files are created by the VAX version of SETUP, called SETUPV.

The basic operation of SETUPV are identical to those of SETUP. The only exception is that the user will be prompted for the names of the four data files to be created. Note that the prompts for the data file names show the "include" file names by which the data files will be accessed in the MERCURY code.

Appendix C contains sample data files for the connectivity and plot data. Refer to Appendix D for a full description of creation of the

connectivity and plot data files using SETUP (SETUPV).

Modifying the Execution Command File for Execution

A sample command file for submission of a MERCURY run on a batch queue is provided in Appendix A, and the list of data descriptions is shown in Table A.1. This command file merely submits the job file for execution on a specific batch queue, names the job, and redirects the log file to the desired location and file name. This file may be modified as desired to match the conditions required.

Modifiying the Job File for Execution

A sample job file is shown in Appendix B, and the list of data descriptions is shown in Figure B.1.

This job file performs the task of compiling (or not compiling) the MERCURY code for a specific flow problem solution on the VAX, then links in the proper data sets and executes the MERCURY code. It also contains the data required by the MERCURY code for execution, so that a separate data file is not required. The basic operation of the job file is as follows:

```
if (compilation is required) then
  assign the "include" file names;
  compile the MERCURY code;
  link the MERCURY code;
else
  do not compile the code;
end if
```

```
if (MERCURY is to be run) then
  assign names for the output, grid and restart files;
  run the MERCURY flow solver;
  deassign the FORTRAN units;
  copy the MERCURY restart file to the desired file;
  delete the MERCURY restart file;
  purge the restart file;
end if;
quit.
```

The job file listed in Appendix B is sufficient to perform the tasks required to achieve flow solutions using MERCURY. It may, however, be customized to perform any additional tasks, as desired. A description of the major portions of the sample job file follows. Note that, although line numbers appear in the job file in Appendix B, they are for reference purposes only, and do not appear in the job file on the VAX. In all cases, the variables which must be changed by the user to match his specific situation are displayed in capital letters in the sample job file.

The first line containing information which the user must modify is on line 1. If the run is an initial run for a flow configuration, the value in the parentheses must be 'COMPILE'. This will instruct the job to compile the MERCURY code, with the appropriate data files, for the particular flow problem under consideration. For subsequent runs, the value should be changed to 'NOCOMPILE' to prevent compilation.

Line 2 contains the variable which is used to determine whether or not the MERCURY code is to be run. This permits the user to compile

executed. If the value of runswitch is 'NORUN', then lines 22 through 33 are not executed. If the value of runswitch is 'RUN' then lines 22 through 33 are executed, and lines 34 through 36 are skipped.

Line 23 assigns the desired output file name with the logical device name 'for006' (FORTRAN unit 6). Replace NOZZLE.OUT with the file name into which you want the output data saved.

Line 24 assigns the desired grid file name with the logical device name 'for004' (FORTRAN unit 4). Replace 'VNOZZBIN.CST' with the file name that corresponds to your grid data file.

Line 25 assigns the desired restart file name to the logical device name 'for001' (FORTRAN unit 1). MERCURY uses FORTRAN unit 1 to read in the restart file which was created from a previous run. If this is the initial run for a flow solution, the VAX will probably give a warning message that the restart file named on this line was not found, but this is not a problem. Replace 'NOZZR.DAT' with the file name that you want to be the restart file.

Line 26 initiates the executable which was created during the linking step (line 11). Therefore, change 'NOZMERC' to match the name of the file which contains the MERCURY executable.

Note that the data required to be read by the flow solver is contained between numbered lines 26 and 27 in the sample job file. This data will be read by MERCURY upon execution of the code. This data is fully explained in the MERCURY User's Manual (CRAY version), Appendix D, with the exception of the following items:

The first three lines contain the names of the output scratch file, the input scratch file and the restart file to be created by

MERCURY, respectively. It is important to use the full pathnames for these files, and to be sure that there is plenty of space available on the specified devices for these files. If the device runs out of room during code execution, the code will crash, and the data from the run will be lost.

Also, the CRAY version of MERCURY permitted the user to specifiy whether or not he wanted to use the solid-state storage device (SSD) for files. The VAX does not support this option, and therefore the input for that option has been removed from the VAX version of MERCURY.

Lines 27 through 29 simply 'deassign' FORTRAN units 1, 4 and 6.

Line 30 copies the 'RESTART.DAT' file, whose name was specified in the flow solver data file and which was created by MERCURY, into the name specified by 'NOZZR.DAT'. Be sure that the file name that you replace 'NOZZR.DAT' with matches the file name that you specified on line 25, and that 'RESTART.DAT' matches the file name specified for the restart file in the flow solver data. Otherwise, the command file will not use the correct file for the restart file on the next run.

Line 31 simply removes the file 'RESTART.DAT' from the disk. This file is no longer needed, since you have copied it to your desired restart file name on line 30 ('NOZZR.DAT').

Line 32 purges the 'NOZZR.DAT' files to reduce the amount of space used by old versions of the restart file. Since these files are quite large, it is important to not keep too many old versions lying around. However, if it is desired to keep at least one old version (for safety's sake), remove this purge command from the file.

Again, the job file may be modified as desired to perform additional

tasks, but the job file given will perform all of the tasks necessary to achieve flow solutions using MERCURY.

NOTE: Due to the long run times to be expected on the VAX, it is best to submit the job file for execution as a batch job by using the sample command file shown in Appendix A.

Sizing the Grid for the VAX

The memory availability on the VAX is not unlimited. Since MERCURY only loads into memory, and operates on, one grid block at a time, the number of blocks which may exist in your grid is virtually unlimited. However, the size of the largest grid block must be less than the amount of memory which is left when you subtract the MERCURY executable code size from the total amount of memory in your machine (that is core memory, not disk storage space).

The following method may be used to determine the approximate size of the largest block which may exist in your grid:

1. Determine the amount of core memory on your VAX (ask your system manager) in words.
2. Determine the approximate size of the executable MERCURY code on your VAX in words.
3. Subtract the executable MERCURY size from the available memory. This will give you net memory.
4. Divide the net memory by 30.0 to arrive at the approximate maximum number of grid points which may exist in the largest grid block. Note that this value is only a very approximate quantity, since the exact MERCURY executable

size will be determined by the maximum grid block size. Some experimenting with the block size for your particular machine may be necessary.

One of the keys to obtaining acceptable MERCURY run times on the VAX is to use block sizes which are as large as possible, and therefore reduce or eliminate the I/O slowdown. Since grid blocks are swapped into and out of core memory after every iteration, reducing the number of blocks is a major step toward lowering the time required for a solution.

Test Run Results

Sample test runs were performed to correlate the VAX version of MERCURY to the CRAY version. The configuration used was internal flow in a converging nozzle at low subsonic velocities. The grid consisted of two blocks, for a total of approximately 51,000 grid points.

The numerical values obtained for the output parameters (e.g. residuals, force and moment coefficients, etc.) were virtually identical between the CRAY and VAX versions (within the roundoff error of the machines).

The CPU times required to achieve the solutions, however, were quite different. Based on the CPU time required to perform a given number of iterations on the same grid and flow configuration, the VAX is approximately 80 times slower than the ASD CRAY X-MP/216 (with the CRAY running under the Cray Operating System (COS)). The VAX used was a VAX 8650.

Section IV

Conclusions

A version of the multiple block Euler code, MERCURY, has been developed for use on the Digital Equipment VAX series of computers. Having the capability of running a detailed, flexible CFD code such as MERCURY, on a machine of the VAX caliber will enhance the flow analysis capabilities of organizations which either do not have access to a CRAY supercomputer, or which must operate in a secure environment.

Sample runs of the VAX version of MERCURY have been made to compare the results against those obtained on the Aeronautical Systems Division (ASD) CRAY at Wright-Patterson Air Force Base. Both single block and multiple block grid configurations have been run and compared. The results obtained from the VAX are identical (within numerical roundoff error) to those obtained on the CRAY.

If there are any questions or problems, contact:

Jed E. Marquart

WRDC/FIMM

WPAFB, OH 45433-6553

(513)255-4052

References

1. Strang, W.Z., "MERCURY User's Manual", AFWAL-TM-88-217, November 1988, (included as Appendix D).

2. Programming in VAX FORTRAN, Software Version 4.0, Digital Equipment Corporation, September 1984.

3. Guide to Using DCL and Command Procedures on VAX/VMS, Software Version 4.0, Digital Equipment Corporation, September 1984.

Appendices

Appendix A

Sample Command File for Batch Execution of MERCURY

```
Line 1:     $ sub/que=BIG$BATCH$1/ -
             log_file=NOZZLE2.LOG/noprint/notify/name=CORNSTALK -
             RUNMERC.JOB
Line 2:     $ exit
```

Table A.1

Summary of Sample Command File Definitions

Line #	Name	Definition
1	BIG$BATCH$1	The name of the batch queue on which the job is to be run.
1	NOZZLE2.LOG	The full pathname and file name of the destination for the log file from the batch job run.
1	CORNSTALK	The name to be assigned to the job on the batch queue.
1	RUNMERC.JOB	The name of the job file which is to be submitted as a batch job on the batch queue.

Appendix B
Sample Flow Solver Job File

```
Line 1:     $ compswitch = "NOCOMPILE"
Line 2:     $ runswitch = "RUN"
Line 3:     $ if compswitch .nes. "COMPILE" then goto nocomp
Line 4:     $  write sys$output " Compiling MERCURY.......please wait"
Line 5:     $   assign PARA.DAT para
Line 6:     $   assign PARAP.DAT parap
Line 7:     $   assign CONN.DAT conn
Line 8:     $   assign PLOT.DAT plot
Line 9:     $   for/object=NOZMERC/list=nozmerc.lst mercvax.for
Line 10:    $    write sys$output " Now linking MERCURY.......please
                 wait"
Line 11:    $   link NOZMERC
Line 12:    $   deassign para
Line 13:    $   deassign parap
Line 14:    $   deassign conn
Line 15:    $   deassign plot
Line 16:    $   goto runcheck
Line 17:    $ nocomp:
Line 18:    $   write sys$output " No compilation was requested"
Line 19:    $   goto runcheck
Line 20:    $ runcheck:
Line 21:    $   if runswitch .nes. "RUN" then goto norun
Line 22:    $    write sys$output " Now executing MERCURY.......please
                 wait (patiently)"
Line 23:    $    assign DISK$AWC_4:[MARQUART.MERCURY]NOZZLE.OUT for006
Line 24:    $    assign   DISK$AWC_4:[MARQUART.MERCURY]VNOZZBIN.CST
                 for004
Line 25:    $    assign DISK$AWC_4:[MARQUART.MERCURY]NOZZR.DAT for001
Line 26:    $    run DISK$AWC_4:[MARQUART.MERCURY]NOZMERC
 'scratch:[marquart]scratcho.dat'
 'scratch:[marquart]scratchi.dat'
 'scratch:[marquart]restart.dat'
 SUBSONIC NOZZLE-2 BLOCKS Mi=0.158 Mo=0.5 01-May-1990
```

```
*****************************
 JOB CONTROL PARAMETERS
*****************************
 START OPTION (0 = INITIAL RUN, 1 = RESTART, 2 = RESTART WITH NEW GRID)
     0
 GRID FORMAT FLAG (0=BINARY, 1=*)
     0
*****************************
 ALGORITHM PARAMETERS
*****************************
 #RUNGE-KUTTA STAGES
     3
 #ITER    #/PLOT    #ITER/(TIME STEP)
  10        -1           -1
  CFL     TIME ACCURATE (0=NO, 1=YES)?
   4.                    0
  VIS2     VIS4      SMOOTHING     DAMPING
  -1.       -1.        -1.           -1.
 TRANSONIC ENTHALPY DAMPING (0=NO, 1=YES)?
   0
*****************************
 FLOW CONDITION PARAMETERS
*****************************
 FREESTREAM MACH NO.     FREESTREAM ALPHA     FREESTREAM BETA
       .158                    0.0                  0.0
    SINK MACH NO.          SINK ALPHA           SINK BETA
       0.5                     -1.                  -1.
   SOURCE MACH NO.        SOURCE ALPHA         SOURCE BETA
       .158                    -1.                  -1.
 REFERENCE MACH NO. FOR CP'S, FORCES AND MOMENTS
      -1.
 RATIO OF SPECIFIC HEATS       SOURCE NPR
      -1.                          -1.
*****************************
 GEOMETRY PARAMETERS
*****************************
```

```
 COORDINATE SYSTEM (0 = FLO57 SYSTEM, 1 = PANEL CODE SYSTEM)
      1
 REFERENCE AREA
   78.5398
 X,Y,Z COORDINATES OF MOMENT REFERENCE POINT
   0.0    0.0    0.0
 REFERENCE LENGTHS FOR MOMENTS ABOUT X-,Y- AND Z-AXIS
  5.000    5.000    5.000
****************************
 BLOCK INFORMATION
****************************
 2
 30 30 28
 31 30 28
Line 27:     $    deassign for001
Line 28:     $    deassign for004
Line 29:     $    deassign for006
Line 30:     $    copy SCRATCH:[MARQUART]RESTART.DAT -
                  DISK$AWC_4:[MARQUART.MERCURY]NOZZR.DAT
Line 31:     $    del SCRATCH:[MARQUART]RESTART.DAT;
Line 32:     $    purge DISK$AWC_4:[MARQUART.MERCURY]NOZZR.DAT
Line 33:     $    goto continue
Line 34:     $ norun:
Line 35:     $    write sys$output " No run was requested"
Line 36:     $    goto exit
Line 37:     $ continue:
Line 38:     $ exit
```

Table B.1

Summary of Sample Job File Definitions

Line #	Name	Definition
1	COMPILE	Flag to indicate that the MERCURY code is to be compiled.
	NOCOMPILE	Flag to indicate that the MERCURY code is not to be compiled.
2	RUN	Flag to indicate that the MERCURY code is to be run.
	NORUN	Flag to indicate that the MERCURY code is not to be run.
5	PARA.DAT	File name for the file containing parametric data (number of grid points, etc.) for the connectivity data set.
6	PARAP.DAT	File name for the file containing parametric data (number of plots) for the plot data set.
7	CONN.DAT	File name for the file containing the connectivity data relating the grid blocks, as well as the boundary conditions.
8	PLOT.DAT	File name for the file containing the plot data (i.e. plot variables, plot direction and frequency, etc.).
9	NOZMERC	File name of the file into which the object code will be saved upon compilation. This will also be the name of the executable code.
11	NOZMERC	The name of the object file which is to be linked. Note that this name must match line 9.
23	NOZZLE.OUT	The full pathname and file name of the file into which the output of the MERCURY run is to be written.
24	VNOZZBIN.CST	The full pathname and file name of the file

		which contains the grid data to be input to MERCURY.
25	NOZZR.DAT	The full pathname and file name of the file which contains the restart data to be input to MERCURY.
26	NOZMERC	The full pathname and file name of the file which contains the executable MERCURY code (must match line 11).
30	RESTART.DAT	The full pathname and file name of the file which contains the restart data (written by MERCURY). Note that this file name must match that given in the flow solver data section of the job file.
	NOZZR.DAT	The full pathname and file name of the file into which the restart file created by MERCURY will be copied (must match line 25).
31	RESTART.DAT	Same as line 30.
32	NOZZR.DAT	Same as line 30.

Appendix C

Sample Connectivity and Plot Data Files

File: PARA.DAT (Part of the Connectivity Data)

```
C
C
      PARAMETER(MAXPT  = 26040)
      PARAMETER(MAXCELL= 28768)
      PARAMETER(MSCELL =   992)
      PARAMETER(MIO    =     1)
      PARAMETER(IBLKS  =     2)
      PARAMETER(MAXDIM =    31)
      PARAMETER(NINT   =     1)
      PARAMETER(MAXINT =     1)
      PARAMETER(MAXBC  =     5)
C
```

File: CONN.DAT (Part of the Connectivity Data)

```
C
C     BLOCK NO.   1 BOUNDARY CONDITIONS
C
      NFIL  (    1) =  0
C
      NUMBC (    1) =  5
C
      MBC   (    1,    1) = 3
      ISIDBC(    1,    1) = 1
C
      IBC(    1,    1,1) =     1
      IBC(    1,    1,2) =    30
      IBC(    1,    1,3) =     1
      IBC(    1,    1,4) =    28
C
```

```
      MBC   (    1,    2) = 5
      ISIDBC(    1,    2) = 3
C
      IBC(    1,    2,1) =     1
      IBC(    1,    2,2) =    30
      IBC(    1,    2,3) =     1
      IBC(    1,    2,4) =    28
C
      MBC   (    1,    3) = 5
      ISIDBC(    1,    3) = 4
C
      IBC(    1,    3,1) =     1
      IBC(    1,    3,2) =    30
      IBC(    1,    3,3) =     1
      IBC(    1,    3,4) =    28
C
      MBC   (    1,    4) = 4
      ISIDBC(    1,    4) = 5
C
      IBC(    1,    4,1) =     1
      IBC(    1,    4,2) =    30
      IBC(    1,    4,3) =     1
      IBC(    1,    4,4) =    30
C
      MBC   (    1,    5) = 6
      ISIDBC(    1,    5) = 6
C
      IBC(    1,    5,1) =     1
      IBC(    1,    5,2) =    30
      IBC(    1,    5,3) =     1
      IBC(    1,    5,4) =    30
C
C     BLOCK NO.   2 BOUNDARY CONDITIONS
C
      NFIL  (    2) =  0
C
```

```
      NUMBC (    2) =  5
C
      MBC   (    2,    1) = 2
      ISIDBC(    2,    1) = 2
C
      IBC(    2,    1,1) =     1
      IBC(    2,    1,2) =    30
      IBC(    2,    1,3) =     1
      IBC(    2,    1,4) =    28
C
      MBC   (    2,    2) = 5
      ISIDBC(    2,    2) = 3
C
      IBC(    2,    2,1) =     1
      IBC(    2,    2,2) =    31
      IBC(    2,    2,3) =     1
      IBC(    2,    2,4) =    28
C
      MBC   (    2,    3) = 5
      ISIDBC(    2,    3) = 4
C
      IBC(    2,    3,1) =     1
      IBC(    2,    3,2) =    31
      IBC(    2,    3,3) =     1
      IBC(    2,    3,4) =    28
C
      MBC   (    2,    4) = 4
      ISIDBC(    2,    4) = 5
C
      IBC(    2,    4,1) =     1
      IBC(    2,    4,2) =    31
      IBC(    2,    4,3) =     1
      IBC(    2,    4,4) =    30
C
      MBC   (    2,    5) = 6
      ISIDBC(    2,    5) = 6
```

```
C
      IBC(    2,    5,1) =     1
      IBC(    2,    5,2) =    31
      IBC(    2,    5,3) =     1
      IBC(    2,    5,4) =    30
C
C
```

File: PARAP.DAT (Part of the Plot Data)

```
C
      PARAMETER(NUMPLOT=    1)
```

File: PLOT.DAT (Part of the Plot Data)

```
C
C
C     PLOT FAMILY NO.   1
C
      ITYPE (    1) =    1
      IBPLOT(    1) =    1
      IPPLOT(    1) =    2
      INPLOT(    1) =   15
      ICP   (    1) =   -1
      IDIR  (    1) =    1
      NSKIP (    1) =    4
```

Appendix D
MERCURY User's Manual (CRAY Version)

TABLE OF CONTENTS

SECTION I
INTRODUCTION

This manual describes the use and operation of MERCURY, an Euler solver developed at WRDC/FIMM. MERCURY is a multiple grid block solver based on Antony Jameson's [1] finite volume, cell centered, four stage Runge-Kutta algorithm with blended second and fourth order artificial dissipation. Implicit residual smoothing [2] and enthalpy damping [1,3] are used to accelerate convergence to a steady-state. Theoretical details and optimization techniques of MERCURY are explained in [4].

Capabilities and Limitations of MERCURY

1. MERCURY solves Euler's inviscid, adiabatic equations of fluid motion. Thus, transport effects (viscosity and thermal diffusion) and heat addition/subtraction effects cannot be treated. The fluid is assumed to obey the perfect gas equation of state. Body forces are neglected.

2. MERCURY is optimized for the Cray X-MP class of computers. It has achieved processing rates of 110 MFLOPS on a 9.5 nanosecond clock cycle Cray X-MP computer and 125 MFLOPS on an 8.5 nanosecond clock cycle Cray X-MP machine. The code's peak rates are probably about 3% higher.

3. MERCURY can handle multi-block grids. There is no limit to the number of grid blocks that can be treated although the maximum number of grid points that can be treated on the ASD Cray X-MP/12 is nine to ten million points. In fact, a 6.64 million point grid of 174 blocks has been solved on the ASD Cray.

4. The largest block treated on the ASD Cray under COS 1.15 contained over 58,800 points. Under COS 1.17, this size decreases to approximately 57,000 points.

5. Each grid block must be organized in a right-hand Cartesian computational coordinate system. Thus, each block will be dimensioned (IL,JL,KL) where by convention, IL is the dimension of the I-index, JL

is the dimension of the J-index and KL is the dimension of the K-index. Each grid block is, thus, ordered into KL planes where each plane is composed of JL lines of IL points.

6. Each block is completely independent of all others with regard to its computational coordinate system.

7. Each block may intersect an arbitrary number of other blocks. Each block may intersect itself.

8. The block intersections, or abutments, are surfaces - not volumes. Thus, block overlap is forbidden.

9. There must be a point-to-point correspondence (zeroth-order continuity) between the participating blocks of an intersection. Higher-order continuity across intersections (slope, curvature, etc.), while not required, improves flow solution accuracy.

10. Any one of the six logical faces of a given block can intersect any face of another block in any fashion as long as point-to-point correspondence is preserved.

11. There are six boundary conditions available: far field, sink, source, solid surface, symmetry and polar singularity. Block intersections are not considered to be boundary conditions. Boundary conditions must be placed on block boundary faces only. An arbitrary number of boundary conditions can be applied to any face of any block.

12. As long as each block is organized as per 5 above, then a block can be of any physical shape. One or more of a block's six logical boundary faces can be collapsed to a line or to a point; in so doing a polar singularity is generated.

13. Boundary conditions and/or intersections are required at every boundary point of every block.

14. The boundary conditions and block intersection information (connectivity) exist in a separate file, called the connectivity file. The information in this file, which is actually FORTRAN code, is merged into MERCURY via the Cray UPDATE utility.

Hardware Requirements

MERCURY is coded in FORTRAN77 specifically for Cray X-MP series computers with DD49 disks and/or a Solid-state Storage Device (SSD). It is impractical to run multiple block solvers on less capable machines because, even with the Cray's fast CPU and input/output (I/O) rates, cases requiring 30 hours of execution time per flow condition are not uncommon. To fully utilize the Cray's power, MERCURY uses several Cray- and site-specific routines. These routines, most of which deal with I/O, are listed in Appendix A.

Additional Software Requirements

The amount of input data, such as the grid and the connectivity datasets, for a multi-block Euler solver is tremendous. This data must meet certain prerequisites if Euler solutions are to be reliably and routinely obtained. Therefore, the reader is strongly encouraged to use QBERT [5] to verify that the grid is right-hand, SETUP (described in Appendix B) to help construct the boundary conditions and connectivity information for the connectivity file and MERCHEK [6] to verify the connectivity file. A Cray procedure file, MERCPROC, contains nearly all the Cray JCL required to operate MERCURY. By using MERCPROC, the user invokes only one command to run MERCURY. MERCPROC is listed in Appendix C.

Software exists to convert a MERCURY flow solution dataset into the format required by the NASA-Ames graphics package, PLOT3D [7]. This software, MERC2PLOT3D and SENDWKS, is invoked in turn to convert a binary solution dataset structured for MERCURY into a Silicon Graphics, Inc. IRIS binary dataset structured for PLOT3D. This software is very easy to use, the user need only supply the names of the datasets to be converted, and is not described in this manual.

SECTION II

RUNNING MERCURY: The Overall Process

(1) A grid is created and saved on the Cray disks.
(2) QBERT is run on the grid to ensure that no left-hand regions exist.
(3) A job file is created. The job file contains: a) the Cray JCL required to run MERCURY and b) $IN, or FORTRAN unit 5; a small input deck containing global variables such as freestream Mach number, angle of attack, etc..
(4) SETUP is used to create the connectivity file and the plot file. MERCURY has the ability to generate various types of plots for diagnostics.
(5) MERCHEK is run to verify the information in the connectivity file.
(6) A flow solution run is initiated when the job file is submitted to the Cray. The grid dataset and a restart file are accessed. A restart file is a file containing the grid coordinate triples and an entire flow solution (density, momenta and energy) for a specific flow condition. If the run is the first for a certain flow condition, then a restart file is unavailable and, hence, MERCURY does not access a restart file in this instance. Either a previously generated MERCURY executable is accessed or a new one is created. Upon job completion, a new restart file is created and saved on the Cray disks. FORTRAN unit 6, or $OUT, is disposed to the front-end. This dataset contains the run's convergence history, integrated force and moment coefficients, the plots specified by the plot file and the Cray dayfile.

The following sections detail each of the above datasets in turn.

SECTION III
MERCURY DATASETS

JOB FILE

The JCL required to run MERCURY is simple and brief. When operating from a VAX front-end, the entire JCL is:

```
JOB,JN=  ,MFL,T= ,SSD= ,CL= .
ACCOUNT,AC= ,APW= ,US= ,UPW= .
*.
OPTION,STAT=OFF.
ACCESS,DN=EXE,PDN=MERCPROC,ID=VAX.
ECHO,OFF=JCL.
CALL,DN=EXE.
ECHO,ON=JCL.
*.
RUN,COMPILE,CID,CED,'CONNECT.DAT','PLOT.DAT',^
GRID,GID,GED,RESTART,RID,RED,PERF,DUMP.
*.
EXIT.
DUMPJOB.
DEBUG,BLOCKS.
/EOF
```

When operating from a CDC front-end, the JCL remains unchanged with the following two exceptions:

```
ACCESS,DN=EXE,PDN=MERCPROC,ID=CDC.
```

and

```
RUN,COMPILE,CID,CED,CONNECT,PLOT,^
GRID,GID,GED,RESTART,RID,RED,PERF,DUMP.
```

Thus, when operating from a given front-end, only two cards will be modified by the user: the JOB and RUN cards.

JOB Card:

Rules of thumb for the time estimate, T= , and SSD requirements, SSD= , are as follows. The time estimate depends upon the #Runge-Kutta stages specified in the algorithm control group of $IN (see below). MERCURY'S typical processing rate on a 9.5 nanosecond Cray X-MP is 10.0 microseconds per grid point per iteration (mspi) for the three stage Runge-Kutta scheme, 12.0 mspi for the four stage scheme and 14.0 mspi for the five stage scheme. Thus, if 1000 iterations are desired on a one million point grid using the four stage scheme, the time estimate should be approximately:

T = .000012 * (# of grid points) * (# of iterations)
T = 12000

The number of SSD sectors requested should be approximately equal to the total number of grid points divided by 35.

Thus, for the above example, the JOB card would be:

JOB,JN=EXAMPLE,MFL,T=12000,SSD=30000,CL= .

When relatively few iterations are desired (<100), the time estimate should be padded an additional 10%. With fewer iterations, normally insignificant operations begin to slow the processing rate. Also, when a grid is to be read, an additional time increment of approximately 100 microseconds per grid point is required to read an ASCII grid dataset. A binary grid dataset requires a time increment of about 50 microseconds per grid point. Additional details of the grid dataset, its format and structure, are given in a subsequent section. If the time request is too short, MERCURY will stop executing and save a new restart file before exceeding the time limit. Lastly, note that the maximum amount of core memory is requested by specifying MFL. This is usually done at WRDC because of the limited memory available on the Cray X-MP/12.

RUN Card:

The RUN command sets in motion the machinery to run MERCURY. The thirteen parameters in the RUN command are described below.

COMPILE (Optional): When specified, the connectivity file, plot file and MERCURY are fetched from the front-end. Through the use of UPDATE, all three datasets are merged into one deck. This deck is then compiled under the CFT77 compiler. At WRDC with the Version 3.0 compiler, the compilation times range from 50 to 600 seconds. The Cray segment loader, SEGLDR, creates an executable which is saved with a permanent dataset name (PDN) of MERCURY, an identifier (ID) of CID and an edition level (ED) of CED. If the COMPILE directive is omitted, an existing executable with ID=CID, ED=CED is accessed.

CID (optional): Specifies the identifier (ID) of the MERCURY executable to be created or accessed.

CED (optional): Specifies the edition (ED) of the MERCURY executable to be created or accessed.

CONNECT.DAT or CONNECT (optional): This parameter is required when COMPILE is specified. Specifies the name of the connectivity file to be fetched from the front-end and included into MERCURY via UPDATE. Whenever a new connectivity file is to be used, COMPILE must be specified to create an executable with the new connectivity information.

PLOT.DAT or PLOT (optional): This parameter is required when COMPILE is specified. Specifies the name of the plot file to be fetched from the front-end and included into MERCURY via UPDATE. Whenever a new plot file is to be used, COMPILE must be specified to create an executable with the new plotting information.

GRID (Required): Specifies the PDN of the grid dataset saved on the Cray.

GID (Optional): Specifies the ID of the grid dataset saved on the Cray. Omit when the grid has no ID.

GED (Optional): Specifies the edition level of the grid dataset saved on the Cray. Omit when the latest edition is desired.

RESTART (Required): Specifies the PDN of the old restart dataset accessed on the Cray. Also specifies the PDN of the new restart file MERCURY creates upon job completion.

RID,RED: Analogous to GID and GED but pertaining to the restart file.

PERF (Optional): When specified, enables the performance monitoring software of the Cray. The performance statistics, such as MFLOPs, are output at the bottom of $OUT.

DUMP (Optional): When specified, the old restart file is deleted from the Cray disks before the new restart is saved.

Let's consider two examples. First, assume a new grid has been saved on the Cray and new connectivity and plot files have been created on the VAX front-end. The grid has a PDN of CAPSULE, while the connectivity and plot files are named CONNCAP.DAT and PLOT.DAT, respectively. The RUN command will be:

```
RUN,COMPILE,B1CAP,2,'CONNCAP.DAT','PLOT.DAT',^
CAPSULE,,,CAPREST,M80A1B3,,,.
```

An executable will be created and saved on the Cray with a permanent dataset name (PDN) of MERCURY, an identifier (ID) of B1CAP and an edition number of two. Upon job completion, a restart file will be saved with a PDN of CAPREST and an ID of M80A1B3.

As a second example, consider a subsequent run of the above case. An executable, therefore, already exists. Let's also delete the old restart file. The RUN command will be:

```
RUN,,B1CAP,2,,,CAPSULE,,,CAPREST,M80A1B3,,,DUMP.
```

By definition, $IN is appended to the JCL below an end-of-file (/EOF). See Figure 1 for a typical job file listing. Note the first line of $IN is a title which is echoed in $OUT.

The parameters in $IN are read by MERCURY at job initiation. These parameters are separated into five major groups. The job control group, consisting of three parameters, tells MERCURY how to access and create datasets. The 11 parameters of the algorithm control group are, with one exception, coefficients required by Jameson's algorithm. The flow condition parameters specify the flow condition to be analyzed. The eight parameters of the geometry definition group specify the coordinate system and certain reference parameters used in the calculation of the integrated force and moment coefficients. The last parameter group is the block information group which MERCURY uses only to perform very simple checks against the information in the grid and restart files. Additionally, when using SETUP to construct the connectivity file, a job file with the correct block information group is required as input to SETUP. See Appendix B for additional information regarding SETUP.

The information in $IN is detailed below. All parameters, unless otherwise noted, are read in a free-field format.

JOB CONTROL GROUP:

Start Option (integer):

0: A grid will be read in and the flow variables will be initialized to the freestream Mach number, angles of attack and sideslip and ratio of specific heats defined in the flow condition group. No restart file is read.

1: A restart file is read in. A grid is not read and the flow variables are not initialized.

```
JOB,JN=SARL,NFL,SSD=10000,T=500.
ACCOUNT,AC=9640262,APW=SIRBAUGH.
*.
OPTION,STAT=OFF.
ACCESS,DN=EXE,PDN=MERCPROC,ID=VAX.
ECHO,OFF=JCL.
CALL,DN=EXE.
ECHO,ON=JCL.
*.
RUN,,SARL,,'CONDCONT.DAT','PLTSARL.DAT',
ARLDIFFGRID,NEWONE80,,SARLREST,M20,,,DUMP.
*.
EXIT.
DUMPJOB.
DEBUG,BLOCKS.
/EOF
 SARL   MACH IN= .6, INF= .05  NPR=1.01  09 JAN 90
*****************************
 JOB CONTROL PARAMETERS
*****************************
 START OPTION (0 = INITIAL RUN, 1 = RESTART, 2 = RESTART WITH NEW GRID)
     1
 I/O DEVICE  (0 = SSD, 1 = DISK)
     0
 GRID FORMAT FLAG (0=BINARY, 1=*)
     0
*****************************
 ALGORITHM PARAMETERS
*****************************
 #RUNGE-KUTTA STAGES
     5
 #ITER   #ITER/PLOT   #ITER/(TIME STEP)
   50       -1            -1
  CFL      TIME ACCURATE (0=NO, 1=YES)?
  10.            0
  VIS2     VIS4     SMOOTHING    DAMPING
  -1.      -1.       -1.          -1.
 TRANSONIC DAMPING (0=NO, 1=YES)?
     0
*****************************
 FLOW CONDITION PARAMETERS
*****************************
 FREESTREAM MACH NO.    FREESTREAM ALPHA     FREESTREAM BETA
      0.05                   0.0                  0.0
    SINK MACH NO.          SINK ALPHA           SINK BETA
      1.10                   -1.                  -1.
   SOURCE MACH NO.        SOURCE ALPHA         SOURCE BETA
      0.80                   -1.                  -1.
 REFERENCE MACH NO. FOR CP'S, FORCES AND MOMENTS
      0.80
 RATIO OF SPECIFIC HEATS      NPR
      -1.                     1.01
*****************************
 GEOMETRY PARAMETERS
*****************************
 COORDINATE SYSTEM (0 = FLO57 SYSTEM, 1 = PANEL CODE SYSTEM)
      1
 REFERENCE AREA
   12867.96
 X,Y,Z COORDINATES OF MOMENT REFERENCE POINT
  0.0    0.0    1050.0
 REFERENCE LENGTHS FOR MOMENTS ABOUT X-,Y- AND Z-AXIS
  64.0   64.0    64.0
*****************************
 BLOCK INFORMATION
*****************************
 8
 50 25 43
 50 25 41
 50 25 41
 50 25 41
 50 25 40
 50 50 17
 50 50 17
 50 50 18
```

Figure 1

2: A restart file is read followed by a read of a new grid. Coordinate triples of the new grid overwrite the old coordinate triples of the restart file. The flow is not initialized. Obviously, the new grid must have precisely the same dimensions as the old grid.

I/O Device (integer):

0: The SSD is to be used. Ensure SSD is specified in the JOB card of the JCL. This option is recommended for multi-block grids due to the SSD's high I/O rate.

1: The disks are to be used. Ensure SSD is not specified in the JOB card. This option is recommended only when treating single block grids.

Grid Format Flag (integer):

0: The grid is an unformatted, binary dataset.

1: The grid is in free field ASCII format.

ALGORITHM CONTROL GROUP:

=Runge-Kutta Stages (integer):

Specifies the number of stages the Runge-Kutta integration scheme is to use. Valid entries are 3, 4, 5 and -1, where -1 invokes the default value. The default is:

3 stage scheme for time accurate solutions.
4 stage scheme when CFL < 10.
5 stage scheme when CFL $\geq$ 10.

where CFL is defined below. The 5 stage scheme tends to be the most efficient scheme of the three except for supersonic flows and unsteady flows. The 3 stage scheme is the best scheme for unsteady flows where the CFL should be +1.0. The four stage scheme is typically the best performer for supersonic flows.

Typical timings for each scheme are:

3 stage = 9.0 to 12.0 microseconds/(grid point - iteration)
4 stage = 11.0 to 15.0 " " "
5 stage = 13.0 to 18.0 " " "

#Iterations (integer):

Specifies number of iterations to be performed during current run.

#Iterations/Plot (integer):

Specifies frequency of generation of diagnostic plots specified in the plot file. As an example, plots will be generated every 100 iterations when this value is 100. Entering a -1 invokes a default value equal to #Iterations. Plots are thus generated only after the last iteration. Entering a value greater than #Iterations disables diagnostic plotting.

#Iterations/(Time Step Calculation) (integer):

Specifies frequency of time step calculation. Works in a fashion analogous to above. Entering -1 invokes the default value of 10 iterations per time step calculation.

CFL (real):

Specifies the Courant-Friedrichs-Lewy number. The following ranges hold for time independent solutions:

3 stage scheme: CFL $\leq$ 7.0
4 stage scheme: CFL $\leq$ 14.0
5 stage scheme: CFL $\leq$ 20.0

whereas for time accurate solutions:

3 stage scheme: CFL $\leq$ 2.0
4 stage scheme: CFL $\leq$ 2.8
5 stage scheme: CFL $\leq$ 4.0

The maximum CFL used on any one problem is a function of the flow conditions and the grid quality. Flows of freestream Mach number near one and/or poor quality grids require a lower CFL.

Time Accurate (integer):

0: Specifies iteration to a steady state solution. Local time stepping, enthaply damping and IRS are all employed as convergence accelerators.

1: Specifies a time accurate solution. Local time stepping, enthalpy damping and IRS are all disabled. The time step is calculated every iteration - regardless of the #Iterations/(Time Step Calculation).

Vis2 (real):

Specifies the coefficient of the second order differences that provide the capability to sharply capture shocks. This coefficient should fall on the interval (0.,.5]. Entering -1. invokes the default value of .333.

Vis4 (real):

Specifies the coefficient of the fourth order differences that provide the dissipation necessary for stability. This coefficient should fall on the interval (0.,4.]. By entering -1., the default value of 2.0 is invoked.

Smoothing (real):

Specifies the coefficient for implicit residual smoothing (IRS). IRS accelerates convergence to a steady-state and is not used for time accurate solutions. This coefficient should fall on the interval [0.,.75]. Entering -1. invokes the default value of MIN(.1*CFL,.75) for time independent solutions and 0. for time accurate solutions.

Damping (real):

Specifies the coefficient of enthalpy damping which also is a convergence accelerator. This coefficient should fall on the interval [0.,.15]. Entering -1. invokes the default value which is 0. unless the freestream Mach number is subsonic, the solution is time independent and the CFL > 2.8. When these conditions are met, the damping coefficient is exponentially related to CFL.

Transonic Damping (integer):

When a transonic flow contains large supersonic pockets, convergence can be accelerated dramatically by turning on the transonic enthalpy damping. Otherwise, the standard enthalpy damping provides better convergence rates.

The CFL, damping, vis2 and vis4 coefficients can have significant effects on convergence rate and solution accuracy. When the freestream Mach number is nearly 1.0, the proper combination of these parameters is critical to obtaining a flow solution. Fortunately, the default values work very well for the majority of cases.

FLOW CONDITION GROUP:

Freestream Mach Number (real):

Specifies the freestream Mach number. This is also the Mach number to which the flow variables are initialized when the start option is zero.

Freestream Angle of Attack (real):

Specifies angle of attack in degrees of vehicle relative to freestream. A positive angle of attack rotates the vehicle's nose upward.

Freestream Angle of Sideslip (real):

Specifies angle of sideslip in degrees of vehicle relative to freestream. A positive angle of sideslip rotates the vehicle such that the wind is in the pilot's right ear.

Sink Mach Number (real):

Specifies the Mach number for the outflow boundary conditions. A value must be input regardless of whether or not a sink boundary condition exists in the problem. If no sink exists in the domain, this value is simply read and not used.

Sink Angle of Attack (real):

Specifies angle of attack in degrees of the sink's flow relative to to the vehicle's coordinate system. Positive 90 degrees orients the sink's flow such that it exits the domain vertically upward. A negative entry enables the default value of zero degrees, i.e. the flow exits the domain parallel to the x-axis.

Sink Angle of Sideslip (real):

Specifies angle of sideslip in degrees of the sink's flow relative to the vehicle's coordinate system. Positive 90 degrees orients the sink's flow such that it exits the domain horizontally to the pilot's left. A negative entry enables the default value of zero degrees.

Source Mach Number (real):

Specifies Mach number to be enforced at inflow boundary conditions. Again, regardless of whether or not a source exists in the problem, a value must be specified.

Source Angle of Attack (real):

Specifies angle of attack in degrees of the source's flow relative to the vehicle's coordinate system. Positive 90 degrees orients the source's flow such that it enters the domain vertically upward. A negative entry enables the default value of zero degrees, i.e. the flow enters the domain parallel to the x-axis.

Source Angle of Sideslip (real):

Specifies angle of sideslip in degrees of the source's flow relative to the vehicle's coordinate system. Positive 90 degrees orients the source's flow such that it enters the domain horizontally from the pilot's right. A negative entry enables the default value of zero degrees.

Reference Mach No. for Cp's, Forces and Moments (real):

Specifies the Mach number used in the calculation of Cp's, forces and moments. The freestream Mach number is the default value and is invoked by a negative entry.

Ratio of Specific Heats (real):

Specifies gamma, the ratio of specific heats. Entering -1. invokes the default value of 1.4.

NPR (real):

Specifies the ratio of the source total pressure to the freestream static pressure to be enforced at the inflow boundary conditions. Entering -1. invokes the default value which is calculated assuming isentropic conditions.

GEOMETRY PARAMETER GROUP:

Coordinate System Flag (integer):

0: Specifies that the geometry is defined in the FLO57 coordinate system. See Figure 2.

1: Specifies that the geometry is defined in the panel code coordinate system. See Figure 2.

Reference Area (real):

Specifies area upon which to base force and moment coefficients.

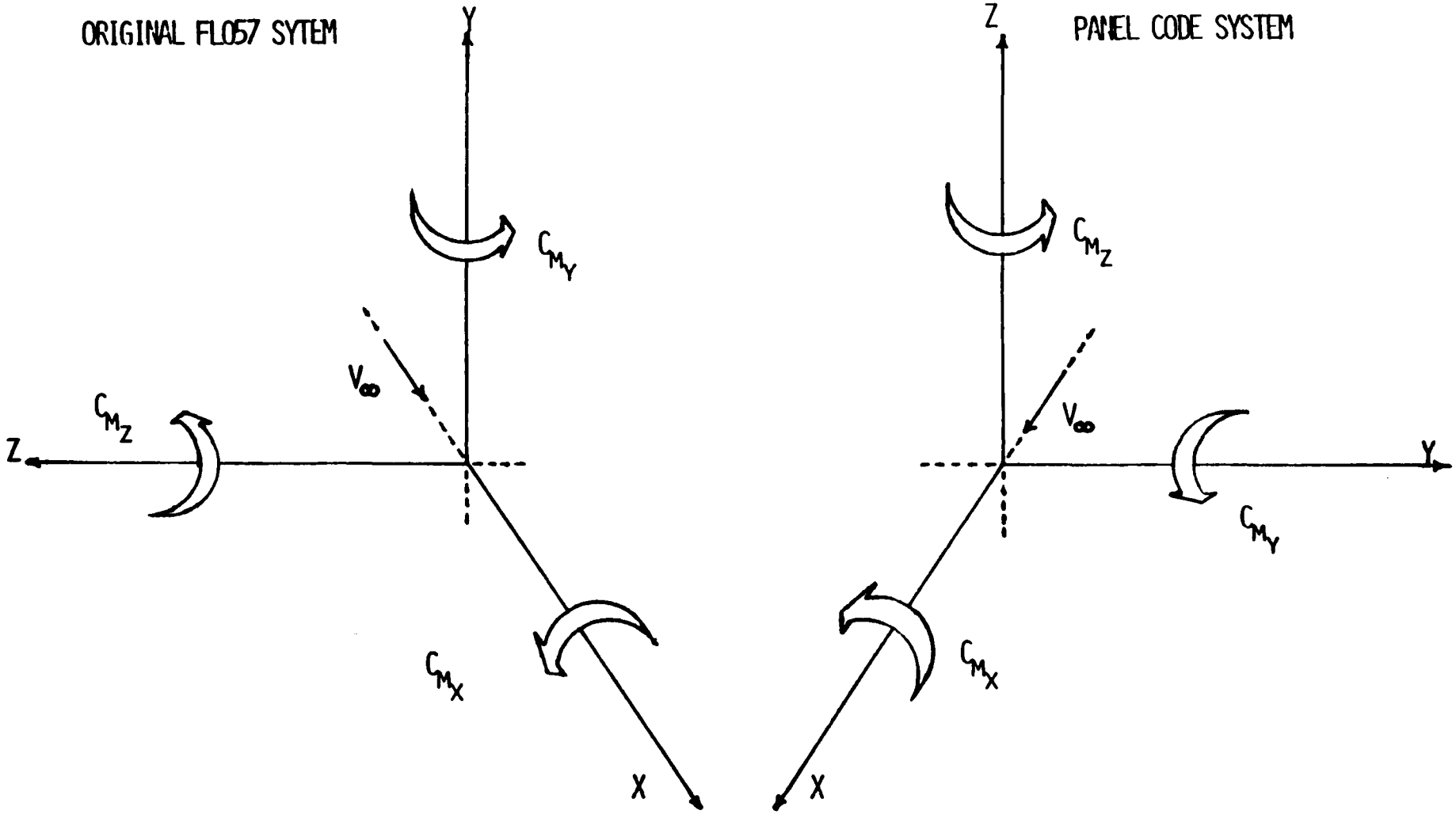
ORIGINAL FL057 SYTEM
Y
C_{M_Y}
V_∞
C_{M_Z}
Z
C_{M_X}
X
PANEL CODE SYSTEM
Z
C_{M_Z}
V_∞
Y
C_{M_Y}
C_{M_X}
X

Figure 2

Moment Reference Point Coordinates (real):

Specifies the x,y,z coordinates of the moment reference point - the point about which all moments are taken.

X-Moment Reference Length (real):

Specifies the reference length upon which to normalize the moment about the x-axis. This moment is the rolling moment in both coordinate systems.

Y-Moment Reference Length (real):

Specifies the reference length upon which to normalize the moment about the y-axis. In the FLO57 coordinate system, this moment is the yawing moment while in the panel code system, it is the pitching moment.

Z-Moment Reference Length (real):

Specifies the reference length upon which to normalize the moment about the z-axis. This moment is the pitching moment in the FLO57 coordinate system and the yawing moment in the panel code system.

BLOCK INFORMATION GROUP:

The line "BLOCK INFORMATION" must be entered such that the "B" in "BLOCK" is placed in column two. This format is required for the proper operation of SETUP. The first number of this group is the number of grid blocks in the grid. Subsequent entries are the I-, J- and K-dimensions of each grid block.

GRID DATASET

The grid must be created and saved on the Cray disks prior to running MERCURY. The grid is read in the following manner:

```
      IF(IFORM.EQ.1) THEN
      READ(4,*) NBLKS
      ELSE
      READ(4)   NBLKS
      END IF
C
      DO 10 IB=1,NBLKS
      IF(IFORM.EQ.1) THEN
      READ(4,*) IL,JL,KL
      ELSE
      READ(4)   IL,JL,KL
      END IF
C
      DO 5 K=1,KL
      DO 5 J=1,JL
      DO 5 I=1,IL
      IF(IFORM.EQ.1) THEN
      READ(4,*) X(I,J,K),Y(I,J,K),Z(I,J,K)
      ELSE
      READ(4)   X(I,J,K),Y(I,J,K),Z(I,J,K)
      END IF
    5 CONTINUE
   10 CONTINUE
```

where:

NBLKS = total number of blocks in the grid
IFORM = grid format flag specified in $IN
IL,JL,KL = the I-, J- and K-dimensions of grid block
X,Y,Z = the Cartesian coordinates of each grid point

and the grid coordinate system, $\hat{i},\hat{j},\hat{k}$ must be right-hand, i.e. $\hat{i}x\hat{j}=\hat{k}$, etc..

RESTART FILE

A restart file is created after each MERCURY job. This dataset contains the coordinate triples of every point as well as the five flow variables at every cell. The dataset is read in the following manner:

```
      READ(1) NBLKS
      READ(1) TIME,LASTCYC
C
      DO 10 IB=1,NBLKS
      READ(1) IL,JL,KL
      NPT = IL*JL*KL
C
      BUFFER IN (1,0) (X(1), X(NPT))
      BUFFER IN (1,0) (Y(1), Y(NPT))
      BUFFER IN (1,0) (Z(1), Z(NPT))
C
      NCELL = (IL+1)*(JL+1)*(KL+1)
      BUFFER IN (1,0) (W1(1), W1(NCELL))
      BUFFER IN (1,0) (W2(1), W2(NCELL))
      BUFFER IN (1,0) (W3(1), W3(NCELL))
      BUFFER IN (1,0) (W4(1), W4(NCELL))
      BUFFER IN (1,0) (W5(1), W5(NCELL))
      IBUF = UNIT(1)
 10   CONTINUE
```

where:

NBLKS	= number of blocks in the grid
TIME	= time; non-zero only for time accurate cases
LASTCYC	= iteration number at restart file creation
IL,JL,KL	= I-, J- and K-dimensions of grid block
X,Y,Z	= Cartesian coordinates of grid point
W1	= density
W2,W3,W4	= x-, y- and z-components of momentum vector
W5	= sum of internal energy and kinetic energy

CONNECTIVITY FILE

The connectivity file is an extremely important file. It contains: 1) all boundary conditions, 2) all grid block intersections (connections) and 3) parameters MERCURY needs to set up its memory. The connectivity file is actually FORTRAN code which, through UPDATE, is merged into the main solver. The connectivity file is clearly segmented into three sections, each performing one of the above functions. All information in the connectivity file is of type integer. A description of each section follows.

BOUNDARY CONDITIONS:

This information, the first section of the connectivity file, is grouped on a block-by-block basis. At the top of each block's boundary conditions is the comment:

```
C     BLOCK NO.   N BOUNDARY CONDITIONS
```

where N is the block's number. There are eight boundary condition parameters per block as described below.

NFIL(N):

Controls the frequency with which the artificial dissipation is calculated in block N. When equal to 0, artificial dissipation is calculated only before the first stage of the Runge-Kutta integration. When equal to 1, the artificial dissipation is calculated before the first and third stages of the four stage and three stage Runge-Kutta schemes, whereas it is calculated before the first two stages of the five stage Runge-Kutta scheme. NFIL is nearly always 0; setting NFIL to 1 may significantly help convergence in poor quality grids.

NUMBC(N):

Number of boundary condition regions on block N describable as L by M surfaces each with a unique boundary condition. Note that block

intersections are not boundary conditions. Boundary conditions, therefore, are applied only on the covering boundary of the global grid. Also, boundary conditions can be applied only on a block's surface - not on its interior. Any number of boundary conditions can be applied to a given block, and multiple conditions can be applied to each face.

MBC(N,NUM):

The NUMth boundary condition applied to block N where $1 \leq NUM \leq NUMBC(N)$. Values for MBC are:

1: far field
2: sink
3: source
4: solid surface
5: symmetry surface
6: polar singularity

where a sink is defined to be a region where flow is leaving the grid domain and a source is defined to be a region where flow is entering the grid domain. Enforcement of options 1, 2 and 3 is via an isenthalpic Riemann Invariant procedure [4] whereas options 4 and 5 are enforced by a flow tangency condition. Boundary condition 6 is required only for proper treatment of the artificial dissipation in any region where the boundary cells have faces of zero area. Again, any boundary condition can be applied anywhere on a block's surface.

ISIDBC(N,NUM):

The face of block N on which the NUMth boundary condition is applied. Values for ISIDBC are:

1: I=1 computational plane
2: I=IL computational plane
3: J=1 computational plane
4: J=JL computational plane

5: K=1 computational plane
6: K=KL computational plane

Hereafter, the I=1 computational plane is termed "face 1", the I=IL plane is termed "face 2", etc.. The next four parameters delimit the region of the face on which a boundary condition is to be enforced. Four parameters are required because, on any given face, two computational indices vary and we must specify the infimum and extremum for each index.

IBC(N,NUM,1) and IBC(N,NUM,2):

Specify the limits of the computational range of the "first" index on face ISIDBC(N,NUM). By convention, the "first" index is defined as follows:

Face	First Index	Second Index
1 (I=1 plane)	J	K
2 (I=IL plane)	J	K
3 (J=1 plane)	I	K
4 (J=JL plane)	I	K
5 (K=1 plane)	I	J
6 (K=KL plane)	I	J

IBC(N,NUM,3) and IBC(N,NUM,4):

Specify the limits of the computational range of the "second" index on face ISIDBC(N,NUM). By convention, the "second" index is defined as above.

For example, consider grid block number 5 whose dimensions are (14,50,40). It has only one boundary condition, a solid surface boundary condition applied to the surface where the index K is everywhere 1 (face 5). Further, the boundary condition is to be enforced over the intervals $1 \le I \le 14$ and $20 \le J \le 50$. See Figure 3. Lastly the artificial dissipation is to be calculated only once per Runge-Kutta step. The boundary condition parameters for this block are thus:

```
      NFIL(5) = 0
C
      NUMBC(5) = 1
C
      MBC(5,1) = 4
      ISIDBC(5,1) = 5
C
      IBC(5,1,1) = 1
      IBC(5,1,2) = 14
      IBC(5,1,3) = 20
      IBC(5,1,4) = 50
```

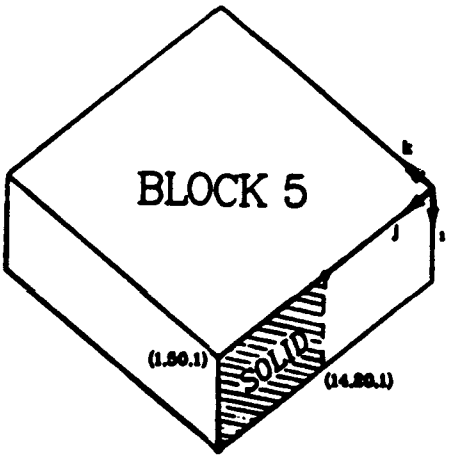

Figure 3

Finally, IBC(N,NUM,1) and IBC(N,NUM,3) are not required to be less than IBC(N,NUM,2) and IBC(N,NUM,4), respectively. The same region in the above example could have been specified with:

```
      IBC(5,1,1) = 14
      IBC(5,1,2) = 1
      IBC(5,1,3) = 50
      IBC(5,1,4) = 20
```

Therefore, there are four equally valid ways to specify the region on which a boundary condition is to be enforced.

BLOCK CONNECTIONS:

Just as the boundary conditions are separated on a block-by-block basis, the connectivity information is broken out on an intersection-by-intersection basis. As previously defined, an intersection is a computational plane (and physical surface) whose grid points are common to two blocks. Each intersection is headed by the comment line:

C INTERSECTION NO. NI PATCHING INDICES

where NI is the intersection number. In a global grid of NINT intersections between all the grid blocks, the following 14 parameters are given for each intersection.

INTER(NI,1) and INTER(NI,2):

The block numbers of the two blocks sharing a common intersection surface where $1 \leq NI \leq NINT$. INTER(NI,1) is the "first" block of intersection NI and INTER(NI,2) is the "second" block of the intersection. It does not matter which block of an intersection is specified as the "first" block.

IPLANE(NI,1) and IPLANE(NI,2):

IPLANE(NI,1) is the face number (of the "first" block) that abuts face number IPLANE(NI,2) of the "second" block of intersection NI. The face numbering convention is the same as that given in the boundary condition section.

INDEX(NI,1) and INDEX(NI,2):

Valid entries for these two parameters depend on the value given IPLANE and are:

IPLANE	INDEX
1 (I=1 plane)	2 or 3
2 (I=IL plane)	2 or 3
3 (J=1 plane)	1 or 3
4 (J=JL plane)	1 or 3

5	(K=1 plane)	1 or 2
6	(K=KL plane)	1 or 2

An INDEX value of 1 denotes the I-direction; 2 denotes the J-direction; 3 denotes the K-direction. The INDEX parameters indicate how the two computational indices varying on each block's abutting face are to be "paired". For example, consider an intersection between blocks 5 and 3. The intersection is part of block 5's fourth face and block 3's sixth face. Thus, the indices I and K of block 5 vary over the intersection whereas indices I and J of block 3 vary over the intersection. The values given for INDEX(NI,1) and INDEX(NI,2) determine whether the I-index of block 5 is colinear with, and hence "paired" with, the I-index or the J-index of block 3. See Figure 4.

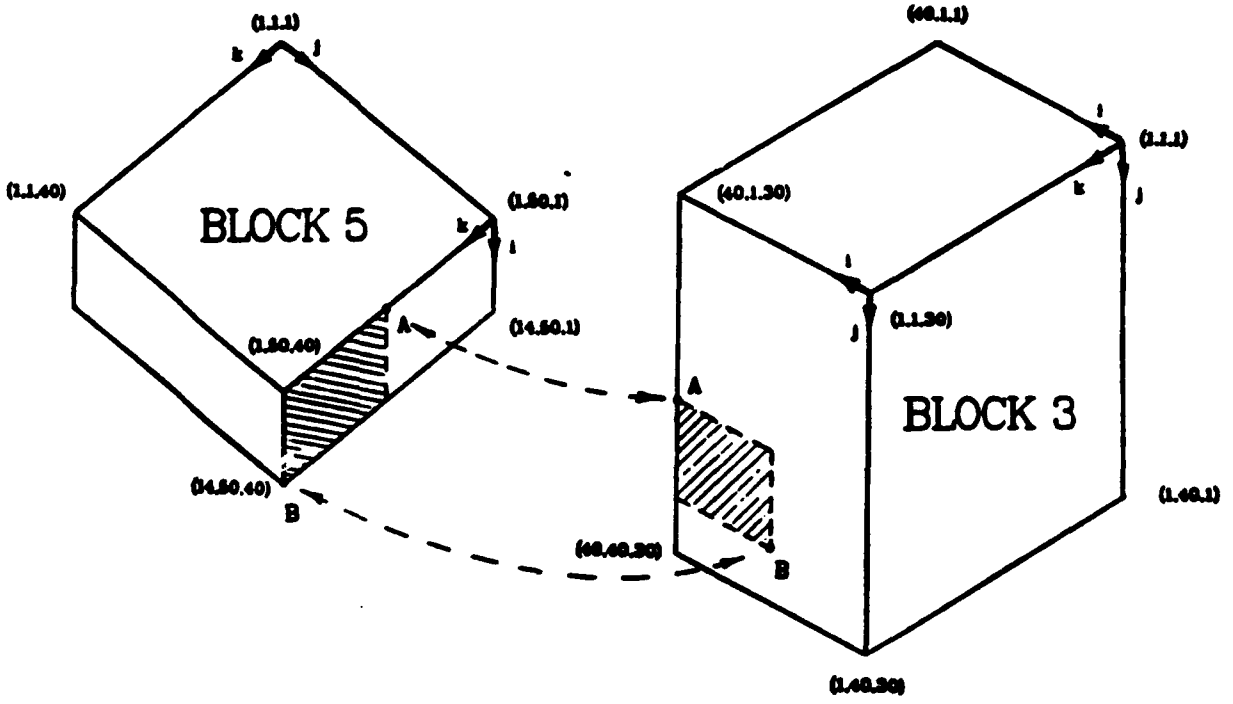

Figure 4

IPNT(NI,1,1) and IPNT(NI,1,2):

The starting and ending coordinates, respectively, of the computational coordinate defined by INDEX(NI,1) delimiting intersection NI on face IPLANE(NI,1) of block INTER(NI,1).

IPNT(NI,1,3) and IPNT(NI,1,4):

The starting and ending coordinates, respectively, of the index not explicitly specified in INDEX(NI,1).

IPNT(NI,2,1) and IPNT(NI,2,2):

Analogous to IPNT(NI,1,1) and IPNT(NI,1,2) but applied to the "second" block of the intersection.

IPNT(NI,2,3) and IPNT(NI,2,4):

Analogous to IPNT(NI,1,3) and IPNT(NI,1,4) but applied to the "second" block of the intersection.

It is important to realize that the four computational coordinates defined by IPNT(NI,1,1-4) are simply the "first" block's coordinates of two diagonally opposite corner points delimiting intersection NI. The coordinates defined by IPNT(NI,2,1-4) define the same two physical points in the "second" block's computational coordinates.

As an example, consider that the third intersection occurs between blocks 5 and 3 who are dimensioned (14,50,40) and (40,40,30), respectively. Block 5, face 4 abuts block 3, face 6. Assume that points A and B are two diagonally opposite corner points delimiting the intersection. See Figure 4. The coordinates of point A in block 5 are (1,50,20) while in block 3 point A is (40,20,30). The coordinates of point B in block 5 are (14,50,40) while in block 3, point B is (20,33,30). Thus, given the fact that a one-to-one correspondence must exist between the points of each block in an intersection, it is obvious that the I-index of block 5 is "paired" with the J-index of block 3 (and the K-index of block 5 is "paired" with the I-index of block 3). It is up to the user to be familiar with the grid and know how the indices in each intersection are "paired". The above example intersection could be specified as follows:

```
      INTER(3,1) = 5
      INTER(3,2) = 3
C
      IPLANE(3,1) = 4
      IPLANE(3,2) = 6
C
      INDEX(3,1) = 1
      INDEX(3,2) = 2
C
      IPNT(3,1,1) = 1
      IPNT(3,1,2) = 14
      IPNT(3,1,3) = 20
      IPNT(3,1,4) = 40
      IPNT(3,2,1) = 20
      IPNT(3,2,2) = 33
      IPNT(3,2,3) = 40
      IPNT(3,2,4) = 20
```

There are 16 possible ways to specify any intersection. This is due to the following facts:

1) There are four choices for point A: any one of the four corners of the intersection may be chosen as point A.

2) There are two choices for the "pairing" indices.

3) There are two choices for the "first" block of the intersection.

Thus, an equally valid specification of the example intersection is:

```
      INTER(3,1) = 5
      INTER(3,2) = 3
C
      IPLANE(3,1) = 4
      IPLANE(3,2) = 6
C
```

```
      INDEX(3,1) = 3
      INDEX(3,2) = 1
C
      IPNT(3,1,1) = 20
      IPNT(3,1,2) = 40
      IPNT(3,1,3) = 1
      IPNT(3,1,4) = 14
      IPNT(3,2,1) = 40
      IPNT(3,2,2) = 20
      IPNT(3,2,3) = 20
      IPNT(3,2,4) = 33
```

PARAMETERS USED TO ESTABLISH MEMORY:

This section occupies the bottom dozen or so lines of the connectivity file. There are nine parameters in this section. All are described below.

MAXCELL:

The greatest number of cells in any grid block. Assuming the blocks are dimensioned (IL,JL,KL), then MAXCELL is the largest value of (IL+1)*(JL+1)*(KL+1) for all grid blocks.

MAXPT:

The greatest number of grid points in any grid block. Using the above convention, MAXPT is the largest value of IL*JL*KL for all grid blocks.

MSCELL:

The greatest number of cells on any computational plane in any block. The number of cells on a constant I-plane is (JL+1)*(KL+1) and similarly for the other faces.

MIO:

The number of 512 word blocks needed to hold the intersection information on the I/O device. Not used for in-core memory management. Users should let SETUP calculate MIO.

MAXDIM:

The greatest number of grid points in any one computational coordinate for all blocks.

IBLKS:

Total number of blocks in the grid.

NINT:

Total number of intersections in the grid.

MAXBC:

The greatest number of boundary conditions applied to any one block.

MAXINT:

The greatest number of intersections in which any one block participates.

MERCURY allocates memory via the following dependence on the above parameters:

```
Memory Req = 22*MAXCELL + 3*MAXPT + 29*MSCELL + 50*MAXDIM + 15*NINT
           + 16*MAXINT + IBLKS*(6*MAXBC + MAXINT + 12) + 1600
```

Of the nine parameters, only IBLKS must always be precisely correct. The parameter NINT must also be precisely correct except when no block connections exist; in this case, NINT equals one. The others may be greater than or equal to what actually exists in the grid. However, valuable memory space is wasted when these parameters are not precisely correct. When SETUP is used to construct the connectivity file, it calculates the precisely correct values for all nine parameters. See Appendix B for additional information on SETUP.

PLOT FILE

The user has the ability to generate Cp, Mach number, relative dynamic pressure, normalized entropy and total pressure loss plots with MERCURY. These plots are not intended for publication purposes but rather for diagnostic purposes. The plots are forced to follow the computational index directions in each block, thus the plots are typically not along a constant physical plane. The plot file is the vehicle by which the user tells MERCURY what plots (Cp, Mach #, etc.) are desired. All entries in the plot file are type integer. Again, SETUP is used to create the plot file.

The plot file is divided into two sections. The first section occupies the first two lines of the plot file and lists a parameter MERCURY requires to set up memory.

NUMPLOT:

Number of plot "families" to be plotted.

The term plot "family" will be defined after describing the second, and major, section of the plot file. The definition will be much easier to understand at that point.

Assume that IPT "families" are to be plotted and $1 \leq IP \leq IPT$. The second section of the plot file contains seven parameters per plot family IP. These parameters are described below.

ITYPE(IP):

1: Denotes that Cp is to be plotted.
2: Denotes that Mach number is to be plotted.
3: Denotes that the ratio of the local dynamic pressure to the freestream dynamic pressure is to be plotted.
4: Denotes that normalized entropy production is to be plotted.
5: Denotes that normalized total pressure loss is to be plotted.

IBPLOT(IP):

The block number from which data is to be plotted.

IPPLOT(IP):

1: Specifies that data is to be plotted from a constant I plane.
2: Specifies that data is to be plotted from a constant J plane.
3: Specifies that data is to be plotted from a constant K plane.

INPLOT(IP):

The value of the above constant. For example if IPPLOT = 1 and INPLOT = 15, then plots are to be taken on the I=15 computational plane.

ICP(IP):

+1: Specifies data is to be taken from the cells just "above" the grid plane defined by IPPLOT and INPLOT. Since MERCURY is a finite volume, cell centered code, data exists not at the grid points but at the grid cell centers. A cell lies "above" the grid plane INPLOT when it lies between the planes INPLOT and INPLOT+1.
-1: Specifies data is to be taken from the cells just "below" the grid plane defined by IPPLOT and INPLOT. A cell lies "below" the grid plane INPLOT when it lies between the planes INPLOT and INPLOT-1.

IDIR(IP):

Defines which computational coordinate is to be the plot abscissa. Valid entries depend on IPPLOT.

IPPLOT	IDIR
1	2,3
2	1,3
3	1,2

NSKIP(IP):

The skip frequency of the plots in the computational direction not specified in IDIR.

As an example, assume that the only plots desired are Cp plots from the J=1 plane of block 4. Thus, NUMPLOT=1. Assume block 4 is dimensioned (IL,JL,KL). We wish to plot the Cp's from the cells just

below the J=1 grid plane, i.e. the ghost cells. We also wish the abscissa of the plots to be the I-direction. Then, since K also varies on a constant J plane, we can get multiple I-direction plots by letting NSKIP<KL-1. If NSKIP=1, then we will obtain a plot "family" of KL-1 plots on the J=1 plane in block 4. Assume we want plots at every third cell in the K-direction. The plot file is then:

```
      PARAMETER(NUMPLOT=1)
C
      ITYPE(1) = 1
      IBPLOT(1) = 4
      IPPLOT(1) = 2
      INPLOT(1) = 1
      ICP(1) = 0
      IDIR(1) = 1
      NSKIP(1) = 3
```

$OUT

The main purposes of $OUT are to provide a convergence history, to provide the plots specified by the plot file and to give the integrated force and moment coefficients. See Appendix D for a partial listing of $OUT. Parameters specified in $IN are also echoed as is a small amount of grid information. With the exception of the convergence history, $OUT is self-explanatory.

Convergence can be measured in several ways. For each block, MERCURY outputs the maximum residual of mass (not density), the indicial location of this maximum, the RMS of the mass residuals, the number of supersonic cells and integrated "forces". The solution is considered converged when the RMS of the mass residual has dropped four to five orders of magnitude, the number of supersonic points is constant and the "forces" are constant. When treating flows where the freestream Mach number is very nearly one, the RMS of the mass residual may not drop four to five orders of magnitude yet the number of supersonic points and the "forces" are constant. Use your judgement to determine whether or not the solution is sufficiently converged.

The term "forces" is somewhat misleading in that it is actually the force divided by the dynamic pressure. These "forces" are accumulated from block to block, thus, the actual "forces" due to a certain block are found by subtracting the "forces" listed at the previous block from those of the block of interest. For example, from Appendix D, the axial "force" due to block 4 alone at iteration 45 is -.622E+04 - (-.143E+05) = +8080.

SECTION IV

GENERAL TIPS ON USING MERCURY

1. The SSD will occasionally be down. In this case, use the disks but be forewarned that the disk I/O wait time will be approximately twice the CPU time.

2. Just as the SSD will be down occasionally, so will one or more of the disks - which, when using disk I/O, causes MERCURY to abort because it expects all the disks to be operational. In this instance, one must inspect the Cray dayfile for the device name(s) of the down disk(s). Then edit the MERCURY source code and search for SUBROUTINE SCRATCH. Very near the top of this subroutine are instructions regarding how to remedy the problem.

3. When using the SSD, ensure adequate SSD space has been requested for the problem at hand. To do this, inspect the very last ten or so lines of the Cray dayfile. You will see two entries: "Job Limit (Sectors)" and "Maximum Concurrent Allocation." The number associated with the job limit is the lowest multiple of 32 that is equal to or greater than the number of SSD sectors requested on the JOB card in the JCL. Make sure this number is at least 512 sectors greater than the number associated with the maximum concurrent allocation. Otherwise, since the Cray grabs SSD space in groups of 512 sectors, you are not guaranteed that the I/O has not spilled over onto the disks. Just to be on the safe side, it is wise to make the job limit at least 1024 sectors greater than the maximum concurrent allocation.

4. Internal flow problems can be quite difficult. The grid must be of very high quality (see [5]) and it can take many iterations to converge the solution. One trick to reduce the number of iterations required for convergence is to set the freestream Mach number equal to the sink Mach number. This causes the flow to be initialized to the sink conditions. This in turn (for some unknown reason) allows a much higher CFL than when the freestream Mach number and the source Mach number are equal.

5. It is wise to attach straight, constant area extensions to both the upstream and downstream ends of internal flow problems. This places the source/sink boundary conditions farther away from the region of interest and, especially at the upstream end, matches the solid wall boundary conditions with the source/sink boundary conditions.

6. MERCURY employs a different artificial dissipation scheme at block boundaries than over block interiors. This boundary dissipation was chosen for its dissipative characteristics and speed of computation. The solution errors due to different dissipation schemes are very small and are swamped by the errors caused by grid spacing changes.

7. Two-dimensional problems can be studied with MERCURY. Stack two 2-D grids together to create a 3-D grid and then apply symmetry boundary conditions on the two walls defined by the 2-D grids to enforce a 2-D solution.

8. The far field boundary should be placed five to ten characteristic lengths away from the vehicle for 3-D problems and 15 to 25 characteristic lengths away for 2-D problems. Characteristic length refers to the longest dimension of the solid body being studied. Closer placement of the far field degrades solution accuracy, while more distant placement needlessly adds grid points.

SECTION V

TROUBLESHOOTING TIPS

"An ounce of prevention is worth a pound of cure." Use QBERT and MERCHEK faithfully to ensure the correctness of the grid and connectivity files. Furthermore, develop a knowledge base of QBERT's cell aspect ratios and grid truncation error estimates correlated with MERCURY's convergence rate and solution accuracy. See [5] for additional details. It is possible to generate a grid of such poor quality (without left-hand regions, however) that given any modest flow gradients, MERCURY will bomb. This problem can be corrected only by regenerating a better grid.

However, even when the grid is of adequate quality and the connectivity file is correct, MERCURY can still abort. When the code aborts, always check the dayfile in $OUT to learn how it aborted. The code will bomb in two ways. The first is hardware problems. Hardware problems can cause any one of four error message types in the dayfile. The messages are "HARDWARE ERROR", "FLOATING POINT OVERFLOW", "SQUARE ROOT OF NEGATIVE NUMBER" and "INVALID ARGUMENT TO MATH LIB ROUTINE". The first two errors are likely due to a problem with an I/O device and the Cray analyst should be notified. The second two errors can be due to hardware problems or due to an instability in the flow solution which grew with time and eventually caused the pressure or density to become negative. Thus, when faced with errors three or four, first assume the hardware is not at fault. Then, try in succession:

(1) Lower the CFL.
(2) Vary the damping, vis2 and vis4 coefficients.
(3) Set NFIL to one in the block(s) where the code bombs.
(4) Input a CFL of 1.0 and request a time accurate solution.
(5) If the above do not work and the flow conditions are hypersonic, you will probably have to give up as MERCURY is not well suited to such cases. If the case is strictly an internal flow problem, ensure the flow desired is physically possible.

When trying the above fixes, always check to see how the code bombs. If the code aborts with a "FLOATING POINT OVERFLOW" one time and with a "SQUARE ROOT OF A NEGATIVE NUMBER" the next time, the real culprit of the problems is hardware-related. Lastly, if none of the above fixes help and hardware problems can be ruled out, the grid is likely at fault. If problems exist, and the grid is known to be of adequate quality, contact:

William Z. Strang
WRDC/FIMM
WPAFB, OH 45433-6553
(513) 255-2481

SECTION VI
CONCLUSION

A multiple grid block Euler code, MERCURY, has been developed using Jameson's algorithm as its kernel. MERCURY enjoys a great deal of flexibility in how blocks may intersect. Any block may intersect an arbitrary number of other blocks, including itself, in any fashion as long as a one-to-one point correspondence is maintained across the intersection. Equally flexible boundary conditions and high I/O and CPU rates have enabled MERCURY to be a versatile and effective performer on many WRDC/FI and ASD studies. For example, the code has been applied to the B-1A escape capsule, numerous Subsonic Aerodynamic Research Research (SARL) wind tunnel flow quality studies, numerous C-135/C-18 MILSTAR/SATCOM configurations, a highly integrated airframe/inlet conceptual fighter, the F-15E, a delta wing run at .6 Mach and 60 degrees angle of attack, numerous subsonic diffusers, a supersonic duct, a converging-diverging nozzle run at various NPR's, two-dimensional airfoils and a ramp run at Mach 15. Freestream Mach numbers have ranged from .005 to 18, steady and unsteady conditions have been studied, grid block sizes have ranged from 56 points to 58,800 points and the number of blocks has ranged from one to 174.

APPENDIX A

Cray-Specific Routines Used In MERCURY

Routine	Purpose
AQOPEN	Open dataset for asynchronous, queued I/O (AQIO)
AQCLOSE	Close AQIO dataset
AQREAD	Read data from AQIO dataset
AQWRITE	Write data to AQIO dataset
AQWAIT	Wait for AQIO completion before proceeding
ASSIGN	Assign a dataset
BTDL	Binary to decimal conversion, left-justified
CLOCK	Supply current wall clock time
DATE	Supply current date
GETPOS	Get dataset position
RELEASE	Release dataset
SETPOS	Set dataset position
TREMAIN	Supply time remaining for job execution

See [8,9,10] for further details.

Site-Specific Code Used In MERCURY

Object	Purpose
PERFMON	Supply MFLOP rating. Located in MERCPROC. See Appendix C.
I/O Device Names	Specify particular I/O devices. Located in subroutine SCRATCH. See Appendix E.

APPENDIX B

Running SETUP

SETUP is invoked on the VAX by typing "RUN SETUP". Since SETUP reads the MERCURY job file, SETUP first requests the job file name. The user is next asked what he wants to do:

1 CREATE GRID CONNECTIVITY FILE

2 CREATE PLOT INFO FILE

3 DO BOTH

Upon answering with a "1", the user is then asked:

IS THIS A RESTART RUN (0=NO, 1=YES)?

Restart in this case does not apply to the MERCURY run, but rather to the particular SETUP session. SETUP allows the user to construct the connectivity file in multiple sessions. Thus, if a portion of a connectivity file already exists, answer affirmatively to this question.

A short cut exists to aid the user in constructing boundary conditions that are applied to entire block faces. By entering a 0 for IBC(N,NUM,1), SETUP calculates the values for IBC(N,NUM,1-4).

Even when no diagnostic plots are desired, a plot file must still be included into MERCURY via UPDATE. Thus, when SETUP asks how many plot families are desired, enter "0". A plot file will be created with NUMPLOT equal to one but without any plotting parameters defined.

All remaining questions should be self-explanatory if the user understands the connectivity file section in this manual. If a problem arises first reference the connectivity file section of this manual.

APPENDIX C

The Cray Procedure File MERCPROC

CDC Version

```
PROC.
RUN,COMP,CID,CED,CONN,PLOT,GRD,GID,GED,RST,RID,RED,PRF,DMP.
*.
IF('&COMP'.EQ.'COMPILE')
*.
FETCH,DN=COMET,SDN=&CONN,TEXT='GET,&CONN.CTASK,ALL.'.
FETCH,DN=CYCLONE,SDN=&PLOT,TEXT='GET,&PLOT.CTASK,ALL.'.
FETCH,DN=CAPRI,SDN=MERCURY,TEXT='ATTACH,MERCURY.CTASK,ALL.'.
*.
UPDATE,P=O,I=COMET:CYCLONE:CAPRI.
CFT77,I=$CPL.
SEGLDR,L=O,CMD='ABS=MERC'.
*.
SAVE,DN=MERC,PDN=MERCURY,ID=&CID,ED=&CED.
RELEASE,DN=COMET:CYCLONE:CAPRI.
RELEASE,DN=$CPL:$NPL:$HLD:$INLINE.
*.
ELSE.
*.
ACCESS,DN=MERC,PDN=MERCURY,ID=&CID,ED=&CED.
*.
ENDIF.
*.
RELEASE,DN=$PROC.
*.
IF('&PRF'.EQ.'PERF')
ACCESS,DN=PERFMON,ID=BNCHMRK,OWN=SYSTEM.
PERFMON,ON=O.
ENDIF.
*.
ACCESS,DN=GRID,PDN=&GRD,ID=&GID,ED=&GED.
ASSIGN,DN=GRID,A=FTO4,LM=200000.
*.
ACCESS,DN=RESTART,PDN=&RST,ID=&RID,ED=&RED,NA.
ASSIGN,DN=RESTART,A=FTO1,LM=200000.
*.
MERC.
*.
IF('&PRF'.EQ.'PERF')
PERFMON,REPORT.
ENDIF.
*.
IF('&DMP'.EQ.'DUMP')
ACCESS,DN=A,PDN=&RST,ID=&RID,ED=&RED,UQ,NA.
DELETE,DN=A,NA.
RELEASE,DN=A.
ENDIF.
*.
SAVE,DN=REST,PDN=&RST,ID=&RID.
*.
ENDPROC.
```

VAX Version

```
PROC.
RUN,COMP,CID,CED,CONN,PLOT,GRD,GID,GED,RST,RID,RED,PRF,DMP.
*.
IF('&COMP'.EQ.'COMPILE')
*.
FETCH,DN=COMET,TEXT=&CONN.
FETCH,DN=CYCLONE,TEXT=&PLOT.
FETCH,DN=CAPRI,TEXT='MERCURY.FOR'.
*.
UPDATE,P=0,I=COMET:CYCLONE:CAPRI.
CFT77,I=$CPL.
SEGLDR,L=0,CMD='ABS=MERC'.
*.
SAVE,DN=MERC,PDN=MERCURY,ID=&CID,ED=&CED.
RELEASE,DN=COMET:CYCLONE:CAPRI.
RELEASE,DN=$CPL:$NPL:$BLD:$INLINE.
*.
ELSE.
*.
ACCESS,DN=MERC,PDN=MERCURY,ID=&CID,ED=&CED.
*.
ENDIF.
*.
RELEASE,DN=$PROC.
*.
IF('&PRF'.EQ.'PERF')
ACCESS,DN=PERFMON,ID=BNCHMRK,OWN=SYSTEM.
PERFMON,ON=0.
ENDIF.
*.
ACCESS,DN=GRID,PDN=&GRD,ID=&GID,ED=&GED.
ASSIGN,DN=GRID,A=FT04,LM=200000.
*.
ACCESS,DN=RESTART,PDN=&RST,ID=&RID,ED=&RED,NA.
ASSIGN,DN=RESTART,A=FT01,LM=200000.
*.
MERC.
*.
IF('&PRF'.EQ.'PERF')
PERFMON,REPORT.
ENDIF.
*.
IF('&DMP'.EQ.'DUMP')
ACCESS,DN=A,PDN=&RST,ID=&RID,ED=&RED,UQ,NA.
DELETE,DN=A,NA.
RELEASE,DN=A.
ENDIF.
*.
SAVE,DN=REST,PDN=&RST,ID=&RID.
*.
ENDPROC.
```

APPENDIX D

Partial Listing of $OUT

```
PROGRAM MERCURY
BLOCKED GRID ANALYSIS OF ARBITRARY FLOW
BY SOLUTION OF UNSTEADY E..ER EQUA..ONS
USING ANTONY JAMESON'S ALGORITHM

QUESTIONS?
CALL WRDC/FIMM AT 54077, 56784 OR 55583

DATE OF SOLUTION IS 01/09/90
TIME OF SOLUTION IS 17:18:23

SARL   MACH IN= .8, INF= .85  NPR=1.01  09 JAN 90
TIME INDEPENDENT SOLUTION
NUMBER OF RUNGE-KUTTA STAGES = 5

FREE STREAM MACH NUMBER    ANG OF ATTACK         YAW
        0.8500              0.0000             0.0000

      SINK MACH NO.        ANG OF ATTACK         YAW
        1.1000              0.0000             0.0000

     SOURCE MACH NO.       ANG OF ATTACK         YAW
        0.6000              0.0000             0.0000

                  GAMMA      SOURCE NPR
                  1.400          1.010

CFL NUMBER     VIS 2       VIS 4   SMOOTHING PARAMETER   DAMPING PARAMETER
  10.0000      0.3333      0.0078         0.7500                0.1210

NUMBER OF ITERATIONS PER TIME STEP IS 10

THERE ARE 8 GRID BLOCKS
BLOCK   1 IS DIMENSIONED    50   25   43
BLOCK   2 IS DIMENSIONED    50   25   41
BLOCK   3 IS DIMENSIONED    50   25   41
BLOCK   4 IS DIMENSIONED    50   25   41
BLOCK   5 IS DIMENSIONED    50   25   40
BLOCK   6 IS DIMENSIONED    50   50   17
BLOCK   7 IS DIMENSIONED    50   50   17
BLOCK   8 IS DIMENSIONED    50   50   18

TOTAL NUMBER OF GRID POINTS IS 387500
```

TOTAL NUMBER OF CELLS IS 422041

FIRST ITERATION	LAST ITERATION
26	75

ITERATION NUMBER 26 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	0.121E+00	8	25	41	0.329E-01	0	-0.487E+04	-0.851E+01	-0.135E+03
2	0.183E+00	50	25	29	0.644E-01	0	-0.937E+04	-0.464E+01	-0.602E+02
3	0.634E-01	19	25	2	0.987E-02	0	-0.186E+05	-0.316E+01	-0.543E+02
4	-0.286E-01	50	25	15	0.133E-01	0	-0.497E+04	0.103E+01	-0.216E+03
5	0.266E-01	50	25	21	0.129E-01	0	-0.195E+05	-0.386E+00	-0.446E+02
6	-0.186E-01	42	50	6	0.787E-02	0	-0.249E+05	-0.427E+00	-0.475E+02
7	0.331E-01	50	11	10	0.423E-02	0	-0.255E+05	-0.552E+00	-0.475E+02
8	0.140E-01	2	23	2	0.266E-02	0	-0.255E+05	-0.552E+00	-0.475E+02

ITERATION NUMBER 30 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	0.849E-01	46	24	41	0.238E-01	0	-0.489E+04	-0.711E+01	-0.112E+03
2	0.139E+00	50	25	27	0.663E-01	0	-0.982E+04	-0.956E+01	-0.151E+03
3	0.927E-01	3	25	2	0.202E-01	0	-0.138E+05	-0.684E+01	-0.109E+03
4	-0.281E-01	50	25	15	0.132E-01	0	-0.533E+04	0.315E+00	-0.247E+03
5	0.253E-01	50	25	16	0.123E-01	0	-0.200E+05	-0.555E+01	-0.106E+03
6	0.150E-01	44	41	2	0.566E-02	0	-0.254E+05	-0.527E+01	-0.105E+03
7	-0.241E-01	2	10	12	0.315E-02	0	-0.260E+05	-0.537E+01	-0.105E+03
8	0.771E-02	26	7	2	0.228E-02	0	-0.260E+05	-0.537E+01	-0.105E+03

ITERATION NUMBER 35 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	0.584E-01	13	24	42	0.153E-01	0	-0.491E+04	-0.417E+01	-0.650E+02
2	0.133E+00	50	25	41	0.613E-01	0	-0.988E+04	-0.780E+01	-0.123E+03
3	0.131E+00	50	25	2	0.344E-01	0	-0.137E+05	-0.522E+01	-0.767E+02
4	-0.242E-01	50	25	15	0.109E-01	0	-0.576E+04	0.552E+01	-0.176E+03
5	0.197E-01	50	25	10	0.984E-02	0	-0.205E+05	-0.300E+01	-0.592E+02
6	0.141E-01	46	41	2	0.452E-02	0	-0.259E+05	-0.253E+01	-0.574E+02
7	-0.249E-01	2	10	16	0.208E-02	0	-0.265E+05	-0.265E+01	-0.571E+02
8	-0.350E-02	50	11	2	0.919E-03	0	-0.265E+05	-0.266E+01	-0.571E+02

ITERATION NUMBER 40 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE

1	0.366E-01	24	24	42	0.944E-02	0	-0.490E+04	-0.382E+01	-0.475E+02
2	0.113E+00	48	26	37	0.586E-01	0	-0.101E+05	-0.764E+01	-0.109E+03
3	0.108E+00	48	25	4	0.428E-01	0	-0.148E+05	-0.636E+01	-0.933E+02
4	-0.192E-01	50	24	13	0.683E-02	0	-0.686E+04	0.780E+01	-0.159E+03
5	0.129E-01	50	24	13	0.629E-02	0	-0.269E+05	-0.428E+01	-0.693E+02
6	0.128E-01	26	43	2	0.364E-02	0	-0.263E+05	-0.384E+01	-0.681E+02
7	-0.177E-01	2	12	16	0.119E-02	0	-0.269E+05	-0.397E+01	-0.680E+02
8	0.395E-02	50	9	2	0.756E-03	0	-0.269E+05	-0.397E+01	-0.680E+02

ITERATION NUMBER 45 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	-0.230E-01	2	9	7	0.582E-02	0	-0.489E+04	-0.172E+01	-0.279E+02
2	0.914E-01	47	25	38	0.387E-01	0	-0.102E+05	-0.637E+01	-0.100E+03
3	0.967E-01	24	25	10	0.478E-01	0	-0.143E+05	-0.390E+01	-0.745E+02
4	0.398E-01	24	25	2	0.122E-01	0	-0.622E+04	0.136E+02	-0.883E+02
5	0.778E-02	50	25	2	0.290E-02	0	-0.211E+05	0.922E-03	-0.151E+02
6	0.110E-01	26	45	2	0.248E-02	0	-0.265E+05	0.631E+00	-0.126E+02
7	0.172E-01	50	10	16	0.736E-03	0	-0.271E+05	0.503E+00	-0.124E+02
8	0.350E-02	50	10	4	0.770E-03	0	-0.271E+05	0.503E+00	-0.124E+02

ITERATION NUMBER 50 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	-0.202E-01	50	25	2	0.484E-02	0	-0.488E+04	-0.108E+01	-0.176E+02
2	0.686E-01	6	24	40	0.257E-01	0	-0.103E+05	-0.568E+01	-0.901E+02
3	0.892E-01	24	25	23	0.487E-01	0	-0.145E+05	-0.881E+00	-0.483E+02
4	0.632E-01	24	25	2	0.265E-01	0	-0.618E+04	0.195E+02	-0.149E+02
5	-0.466E-02	26	22	18	0.148E-02	0	-0.210E+05	0.516E+01	0.515E+02
6	0.850E-02	2	47	2	0.173E-02	0	-0.264E+05	0.582E+01	0.550E+02
7	0.133E-01	2	11	15	0.521E-03	0	-0.270E+05	0.570E+01	0.552E+02
8	0.290E-02	2	10	7	0.598E-03	0	-0.270E+05	0.570E+01	0.552E+02

ITERATION NUMBER 55 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	-0.161E-01	2	25	2	0.328E-02	0	-0.487E+04	-0.102E+01	-0.161E+02
2	0.487E-01	24	24	40	0.153E-01	0	-0.103E+05	-0.445E+01	-0.742E+02
3	0.921E-01	50	25	33	0.479E-01	0	-0.146E+05	0.190E+01	-0.171E+02
4	0.834E-01	2	25	2	0.403E-01	0	-0.000E+04	0.240E+02	0.483E+02
5	0.753E-02	16	2	2	0.290E-02	0	-0.209E+05	0.103E+02	0.120E+03
6	0.700E-02	8	50	2	0.169E-02	0	-0.262E+05	0.109E+02	0.123E+03
7	0.771E-02	2	12	14	0.566E-03	0	-0.268E+05	0.107E+02	0.123E+03
8	0.200E-02	2	10	9	0.447E-03	0	-0.268E+05	0.107E+02	0.123E+03

ITERATION NUMBER 60 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	-0.142E-01	2	25	2	0.330E-02	0	-0.486E+04	-0.107E+01	-0.172E+02
2	0.315E-01	24	24	40	0.782E-02	0	-0.103E+05	-0.315E+01	-0.555E+02
3	0.928E-01	50	25	41	0.438E-01	0	-0.147E+05	0.294E+01	0.398E+01
4	0.945E-01	50	25	2	0.510E-01	0	-0.574E+04	0.255E+02	0.804E+02
5	0.170E-01	17	2	2	0.457E-02	0	-0.206E+05	0.132E+02	0.169E+03
6	0.641E-02	44	50	2	0.193E-02	0	-0.259E+05	0.135E+02	0.170E+03
7	-0.772E-02	2	9	14	0.602E-03	0	-0.206E+05	0.134E+02	0.170E+03
8	0.227E-02	2	10	12	0.373E-03	0	-0.206E+05	0.134E+02	0.170E+03

ITERATION NUMBER 65 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	-0.110E-01	2	25	2	0.365E-02	0	-0.484E+04	-0.116E+01	-0.190E+02
2	0.199E-01	24	24	40	0.484E-02	0	-0.102E+05	-0.193E+01	-0.364E+02
3	0.917E-01	50	25	41	0.389E-01	0	-0.148E+05	0.217E+01	0.851E+01
4	0.960E-01	50	25	14	0.576E-01	0	-0.543E+04	0.239E+02	0.736E+02
5	0.283E-01	16	2	2	0.798E-02	0	-0.202E+05	0.142E+02	0.192E+03
6	0.451E-02	50	50	2	0.192E-02	0	-0.258E+05	0.142E+02	0.191E+03
7	-0.664E-02	2	10	14	0.773E-03	0	-0.262E+05	0.141E+02	0.191E+03
8	0.245E-02	2	9	17	0.344E-03	0	-0.262E+05	0.141E+02	0.191E+03

ITERATION NUMBER 70 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	-0.106E-01	50	25	2	0.436E-02	0	-0.482E+04	-0.121E+01	-0.200E+02
2	0.132E-01	24	24	41	0.488E-02	0	-0.102E+05	-0.164E+01	-0.203E+02
3	0.816E-01	50	25	41	0.328E-01	0	-0.149E+05	0.242E+00	-0.778E+00
4	0.976E-01	50	25	16	0.595E-01	0	-0.512E+04	0.196E+02	0.282E+02
5	0.391E-01	16	2	2	0.139E-01	0	-0.200E+05	0.136E+02	0.186E+03
6	-0.362E-02	47	38	2	0.137E-02	0	-0.253E+05	0.128E+02	0.183E+03
7	-0.486E-02	2	11	14	0.739E-03	0	-0.259E+05	0.126E+02	0.183E+03
8	-0.261E-02	2	7	17	0.372E-03	0	-0.259E+05	0.126E+02	0.183E+03

ITERATION NUMBER 75 TIME= 0.

BLOCK	DM/DT MAX	I	J	K	DM/DT RMS	SUPERSONIC CELLS	AXIAL FORCE	NORMAL FORCE	SIDE FORCE
1	-0.128E-01	11	25	39	0.513E-02	0	-0.480E+04	-0.122E+01	-0.201E+02
2	-0.127E-01	50	25	6	0.502E-02	0	-0.102E+05	-0.550E+00	-0.933E+01
3	0.681E-01	50	25	41	0.271E-01	0	-0.149E+05	-0.192E+01	-0.177E+02
4	0.964E-01	50	25	16	0.577E-01	0	-0.484E+04	0.142E+02	-0.396E+02

5	0.496E-01	50	2	2	0.213E-01	0	-0.190E+05	0.105E+02	0.156E+03
6	0.595E-02	23	20	2	0.136E-02	0	-0.252E+05	0.992E+01	0.151E+03
7	-0.304E-02	2	12	13	0.514E-03	0	-0.250E+05	0.977E+01	0.151E+03
8	-0.218E-02	2	8	17	0.391E-03	0	-0.250E+05	0.977E+01	0.151E+03

MACH PLOT FAMILY 1 BLOCK 1

X	Y	Z	MACH	
1044.62	-1.33	-0.00	0.2293	•
1048.21	-1.33	-0.00	0.2365	•
1051.79	-1.27	-0.00	0.2046	•
1055.38	-1.21	-0.00	0.1872	•
1058.97	-1.21	-0.00	0.1625	•
1062.55	-1.21	-0.00	0.1465	•
1066.14	-1.21	-0.00	0.1312	•
1069.73	-1.21	-0.00	0.1192	•
1073.32	-1.21	-0.00	0.1090	•
1076.90	-1.21	-0.00	0.1025	•
1080.49	-1.21	-0.00	0.0969	•
1084.08	-1.21	-0.00	0.0926	•
1087.67	-1.21	-0.00	0.0893	•
1091.25	-1.21	-0.00	0.0867	•
1094.84	-1.21	-0.00	0.0847	•
1098.43	-1.21	-0.00	0.0831	•
1102.01	-1.21	-0.00	0.0818	•
1105.60	-1.21	-0.00	0.0807	•
1109.19	-1.21	-0.00	0.0798	•
1112.78	-1.21	-0.00	0.0790	•
1116.36	-1.21	-0.00	0.0783	•
1119.95	-1.21	-0.00	0.0777	•
1123.54	-1.21	-0.00	0.0772	•
1127.12	-1.22	-0.00	0.0767	•
1130.71	-1.22	-0.00	0.0762	•
1134.30	-1.22	-0.00	0.0758	•
1137.89	-1.22	-0.00	0.0754	•
1141.47	-1.22	-0.00	0.0750	•
1145.06	-1.22	-0.00	0.0747	•
1148.65	-1.22	-0.00	0.0743	•
1152.23	-1.22	-0.00	0.0740	•
1155.82	-1.22	-0.00	0.0737	•
1159.41	-1.22	-0.00	0.0735	•
1163.00	-1.22	-0.00	0.0732	•
1166.58	-1.22	-0.00	0.0730	•
1170.17	-1.22	-0.00	0.0728	•

2744.62	47.98	-496.57	0.6499	•
2745.39	49.09	-508.48	0.6499	•
2746.14	50.22	-520.41	0.6500	•

MACH PLOT FAMILY 11 BLOCK 8

X	Y	Z	MACH	
2653.57	-3.64	-0.47	0.6500	•
2655.06	-11.01	-1.41	0.6500	•
2657.52	-18.64	-2.40	0.6500	•
2660.52	-26.66	-3.43	0.6500	•
2663.84	-35.09	-4.52	0.6500	•
2667.32	-43.98	-5.66	0.6500	•
2670.87	-53.05	-6.85	0.6500	•
2674.41	-62.50	-8.07	0.6500	•
2677.89	-72.21	-9.33	0.6499	•
2681.29	-82.15	-10.62	0.6499	•
2684.57	-92.28	-11.94	0.6499	•
2687.74	-102.58	-13.27	0.6499	•
2690.78	-113.03	-14.63	0.6499	•
2693.70	-123.62	-16.00	0.6499	•
2696.49	-134.31	-17.39	0.6499	•
2699.17	-145.11	-18.79	0.6499	•
2701.72	-155.99	-20.20	0.6499	•
2704.16	-166.94	-21.62	0.6499	•
2706.49	-177.96	-23.05	0.6499	•
2708.72	-189.04	-24.48	0.6499	•
2710.85	-200.18	-25.92	0.6499	•
2712.89	-211.35	-27.36	0.6499	•
2714.85	-222.57	-28.81	0.6499	•
2716.72	-233.81	-30.26	0.6499	•
2718.51	-245.09	-31.71	0.6499	•
2720.22	-256.39	-33.16	0.6499	•
2721.87	-267.71	-34.62	0.6499	•
2723.45	-279.04	-36.07	0.6499	•
2724.97	-290.39	-37.53	0.6499	•
2726.43	-301.75	-38.99	0.6499	•
2727.83	-313.11	-40.45	0.6499	•
2729.18	-324.47	-41.90	0.6499	•
2730.48	-335.83	-43.36	0.6499	•
2731.73	-347.18	-44.82	0.6499	•
2732.94	-358.54	-46.27	0.6499	•
2734.10	-369.88	-47.73	0.6499	•
2735.22	-381.22	-49.18	0.6499	•
2736.31	-392.55	-50.64	0.6499	•

2737.36	-403.88	-52.09	0.0499	•
2738.37	-415.20	-53.54	0.0499	•
2739.35	-426.53	-55.00	0.0499	•
2740.30	-437.86	-56.46	0.0499	•
2741.22	-449.21	-57.92	0.0499	•
2742.10	-460.59	-59.38	0.0499	•
2742.96	-472.00	-60.85	0.0499	•
2743.79	-483.48	-62.33	0.0499	•
2744.60	-495.05	-63.82	0.0499	•
2745.38	-506.74	-65.33	0.0500	•
2746.13	-518.59	-66.86	0.0500	•

****

****

CONFIGURATION CHARACTERISTICS

FREE STREAM MACH NO.	ALPHA	BETA
0.0500	0.0000	0.0000

REFERENCE MACH NO.	REFERENCE AREA
0.0000	12067.96

MOMENT REF LENGTHS			MOMENT REF POINT		
X	Y	Z	X	Y	Z
64.00	64.00	64.00	0.00	0.00	1050.00

****

COEFFICIENTS WITH RESPECT TO THE
WIND-AXIS SYSTEM

CL	CD INVISCID	L/D	CY
0.0000	-2.0004	0.00	0.0117

CM PITCH	CM YAW	CM ROLL
32.0007	0.3077	0.1021

****

COEFFICIENTS WITH RESPECT TO THE
STABILITY-AXIS SYSTEM

```
          CL      CD INVISCID       L/D         CY
        0.0000      -2.0004        0.00       0.0117

              CM PITCH      CM YAW       CM ROLL
              32.0967       0.3077       0.1921

*************************************************************************

             COEFFICIENTS WITH RESPECT TO THE
                   BODY-AXIS SYSTEM

          CL      CD INVISCID       L/D         CY
        0.0000      -2.0004        0.00       0.0117

              CM PITCH      CM YAW       CM ROLL
              32.0967       0.3077       0.1921

*************************************************************************


 CPU RATE: 4.85 SECONDS/ITER
           12.51 MICROSECONDS/(PT-ITER)
           11.48 MICROSECONDS/(CELL-ITER)
           2.502 MICROSECONDS/(PT-STAGE)
           2.292 MICROSECONDS/(CELL-STAGE)

1
  17:18:21.9614       0.0000    CSP             CRAY X-MP SERIAL-418/48 1.17UP1-A    WPAFB/ASD ISTC     01/09/90
  17:18:21.9617       0.0000    CSP
  17:18:21.9620       0.0000    CSP             CRAY OPERATING SYSTEM          COS 1.17  ASSEMBLY DATE 10/27/89
  17:18:21.9623       0.0000    CSP
  17:18:21.9625       0.0000    CSP
  17:18:21.9644       0.0000    CSP     JOB,JN=SARL,MFL,SSD=10000,T=500.
  17:18:22.0290       0.0011    CSP     ACCOUNT,AC=,APW=.
  17:18:22.0294       0.0012    CSP
  17:18:22.0297       0.0012    CSP     ......................................................................
  17:18:22.0300       0.0012    CSP
  17:18:22.4102       0.0079    EXP     *.
  17:18:22.4130       0.0079    CSP     OPTION,STAT=OFF.
  17:18:22.4145       0.0084    CSP     ACCESS,DN=EXE,PDN=MERCPROC,ID=VAX.
  17:18:22.4645       0.0084    PDM     PD000 - PDN = MERCPROC          ID = VAX       ED =    3  OWN = D040262
  17:18:22.4650       0.0084    PDM     PD000 - ACCESS  COMPLETE, SIZE =       1
  17:18:22.4664       0.0085    CSP     ECHO,OFF=JCL.
  17:18:22.6212       0.0119  1 CSP     CS017 - (RE)CREATING LIBARARY: SPROC
```

```
17:18:22.6768      0.0147    EXP     *.
17:18:22.6959      0.0158    CSP     RUN,,SARL,,'CONDCMNT.DAT','PLTSARL.DAT',
17:18:22.6964      0.0158    CSP     SARLDIFFGRID,NEWONE00,,SARLREST,M20,,,DUMP.
17:18:22.7974      0.0254  1 EXP     *.
17:18:22.7983      0.0255  1 CSP     IF(''.EQ.'COMPILE')
17:18:22.8003      0.0257  1 CSP     ELSE.
17:18:22.8008      0.0258  1 EXP     *.
17:18:22.8020      0.0261  1 CSP     ACCESS,DN=MERC,PDN=MERCURY,ID=SARL,ED=.
17:18:22.8278      0.0262  1 PDM     PD000 - PDN = MERCURY          ID = SARL      ED =    2  OWN = D840262
17:18:22.8281      0.0262  1 PDM     PD000 - ACCESS  COMPLETE, SIZE =    1416
17:18:22.8289      0.0262  1 EXP     *.
17:18:22.8298      0.0262  1 CSP     ENDIF.
17:18:22.8303      0.0263  1 EXP     *.
17:18:22.8312      0.0263  1 CSP     RELEASE,DN=$PROC.
17:18:22.8325      0.0264  1 EXP     *.
17:18:22.8333      0.0264  1 CSP     IF(''.EQ.'PERF')
17:18:22.8342      0.0266  1 CSP     ENDIF.
17:18:22.8347      0.0266  1 EXP     *.
17:18:22.8359      0.0271  1 CSP     ACCESS,DN=QRID,PDN=SARLDIFFQRID,ID=NEWONE00,ED=.
17:18:22.8587      0.0271  1 PDM     PD000 - PDN = SARLDIFFQRID     ID = NEWONE00  ED =    2  OWN = D840262
17:18:22.8589      0.0271  1 PDM     PD000 - ACCESS  COMPLETE, SIZE =    3034
17:18:22.8604      0.0272  1 CSP     ASSIGN,DN=QRID,A=FT04,LM=200000.
17:18:22.8624      0.0274  1 EXP     *.
17:18:22.8638      0.0278  1 CSP     ACCESS,DN=RESTART,PDN=SARLREST,ID=M20,ED=,NA.
17:18:22.8997      0.0278  1 PDM     PD000 - PDN = SARLREST         ID = M20       ED =    1  OWN = D840262
17:18:22.9000      0.0278  1 PDM     PD000 - ACCESS  COMPLETE, SIZE =    6413
17:18:22.9018      0.0279  1 CSP     ASSIGN,DN=RESTART,A=FT01,LM=200000.
17:18:22.9033      0.0281  1 EXP     *.
17:18:22.9228      0.0283  1 CSP     MERC.
17:27:21.1781    243.7938  1 USER    UT010 - STOP          In MERCURY
17:27:21.1785    243.7938  1 EXP     *.
17:27:21.1831    243.7939  1 CSP     IF(''.EQ.'PERF')
17:27:21.1839    243.7940  1 CSP     ENDIF.
17:27:21.1844    243.7941  1 EXP     *.
17:27:21.1854    243.7941  1 CSP     IF('DUMP'.EQ.'DUMP')
17:27:21.1872    243.7946  1 CSP     ACCESS,DN=A,PDN=SARLREST,ID=M20,ED=,UQ,NA.
17:27:21.2702    243.7947  1 PDM     PD000 - PDN = SARLREST         ID = M20       ED =    1  OWN = D840262
17:27:21.2707    243.7947  1 PDM     PD000 - ACCESS  COMPLETE, SIZE =    6413
17:27:21.2720    243.7949  1 CSP     DELETE,DN=A,NA.
17:27:21.3033    243.7949  1 PDM     PD000 - PDN = SARLREST         ID = M20       ED =    1  OWN = D840262
17:27:21.3036    243.7949  1 PDM     PD000 - DELETE  COMPLETE, SIZE =    6413
17:27:21.3052    243.7950  1 CSP     RELEASE,DN=A.
17:27:21.3073    243.7951  1 CSP     ENDIF.
17:27:21.3078    243.7951  1 EXP     *.
```

```
17:27:21.3693     243.7964  1 CSP     SAVE,DN=REST,PDN=SARLREST,ID=M20.
17:27:21.4199     243.7965  1 PDM     PD000 - PDN = SARLREST          ID = M20       ED =    1  OWN = D840262
17:27:21.4201     243.7965  1 PDM     PD000 - SAVE    COMPLETE, SIZE =    6413
17:27:21.4209     243.7965  1 EXP     *.
17:27:21.4385     243.7965    EXP     *.
17:27:21.4431     243.7966    CSP     EXIT.
17:27:21.4434     243.7965    CSP     END OF JOB
17:27:21.4632     243.7966    CSP
17:27:21.4634     243.7966    CSP
17:27:21.5210     243.7967    USER        JOB NAME -                          SARL
17:27:21.5217     243.7967    USER        USER NUMBER -                       D840262
17:27:21.5221     243.7967    USER        JOB CLASS -                           SMALL
17:27:21.5224     243.7967    USER        JOB SEQUENCE NUMBER -                  1467
17:27:21.5227     243.7967    USER
17:27:21.5231     243.7968    USER        TIME EXECUTING IN CPU -             0000:04:03.7958
17:27:21.5235     243.7968    USER        TIME WAITING TO EXECUTE -           0000:04:38.4317
17:27:21.5238     243.7968    USER        TIME WAITING FOR I/O -              0000:00:11.0779
17:27:21.5241     243.7968    USER        TIME WAITING SEMAPHORE -            0000:00:00.0000
17:27:21.5245     243.7968    USER        TIME WAITING IN INPUT QUEUE -       0000:00:00.0469
17:27:21.5248     243.7969    USER        MEMORY * CPU TIME (MWDS*SEC) -           394.27564
17:27:21.5251     243.7969    USER        MEMORY * I/O WAIT TIME (MWDS*SEC) -       17.12455
17:27:21.5255     243.7969    USER        MEMORY * SEM WAIT TIME (MWDS*SEC) -        0.00000
17:27:21.5258     243.7969    USER        MINIMUM JOB SIZE (WORDS) -              56832
17:27:21.5261     243.7969    USER        MAXIMUM JOB SIZE (WORDS) -            1628160
17:27:21.5266     243.7969    USER        MINIMUM FL (WORDS) -                    52736
17:27:21.5271     243.7969    USER        MAXIMUM FL (WORDS) -                  1628552
17:27:21.5274     243.7969    USER        MINIMUM JTA (WORDS) -                    4096
17:27:21.5277     243.7969    USER        MAXIMUM JTA (WORDS) -                    5632
17:27:21.5281     243.7969    USER        DISK SECTORS MOVED -                    14477
17:27:21.5284     243.7969    USER        FSS SECTORS MOVED -                   1007827
17:27:21.5287     243.7969    USER        USER I/O REQUESTS -                      3664
17:27:21.5290     243.7969    USER        USER I/O SUSPENSIONS -                   4635
17:27:21.5294     243.7969    USER        OPEN CALLS -                               24
17:27:21.5297     243.7969    USER        CLOSE CALLS -                               5
17:27:21.5300     243.7969    USER        MEMORY RESIDENT DATASETS -                  0
17:27:21.5303     243.7969    USER        TEMPORARY DATASET SECTORS USED -        15255
17:27:21.5307     243.7969    USER        PERMANENT DATASET SECTORS ACCESSED -    17375
17:27:21.5310     243.7969    USER        PERMANENT DATASET SECTORS SAVED -        6413
17:27:21.5320     243.7981    USER        SECTORS RECEIVED FROM FRONT END -           0
17:27:21.5323     243.7981    USER        SECTORS QUEUED TO FRONT END -               0
17:27:21.5326     243.7981    USER
17:27:21.5333     243.7981    USER        GENERIC RESOURCE NAME -                   SSD
17:27:21.5336     243.7981    USER        DEVICE TYPE -                            DISK
17:27:21.5339     243.7981    USER        JOB LIMIT (SECTORS) -                   16016
```

```
17:27:21.5342   243.7961   USER   MAXIMUM CONCURRENT ALLOCATION -                    8960
17:27:21.5346   243.7961   USER   ALLOCATION INTEGRAL (MEGASECTOR*SEC)            4.76247
17:27:21.5349   243.7961   USER   SECTORS MOVED -                                 1007827
17:27:21.5353   243.7962   USER   JOB CLASS                              SMALL
17:27:21.5356   243.7962   USER   JOB COST (APPROXIMATE) -   $   135.84
17:27:21.5359   243.7962   USER   TOTAL SBUS USED -                                    0
```

APPENDIX E

MERCURY Subroutines

MERCURY'S subroutines are functionally grouped as follows:

Main Program
Routines to set up the problem
Flow solution routines
Boundary condition routines
I/O routines
Convergence measurement routines
Miscellaneous output routines

Within each group, the subroutines are alphabetized.

MAIN PROGRAM

PROGRAM MERCURY - Coordinates efforts of all subroutines; reads $IN, writes $OUT, calculates force and moment coefficients.

ROUTINES TO SET UP THE PROBLEM

SUBROUTINE INIT - Initializes flow variables.

SUBROUTINE KONNECT - Holds connectivity information and boundary conditions; sets addresses for random access I/O.

SUBROUTINE METRIC - Calculates the cell volume reciprocals for time accurate solutions only. Calculates unit normals on the six faces of every block.

SUBROUTINE SCRATCH - Establishes random access scratch files on I/O device(s).

SUBROUTINE SETUP - Coordinates routines involved with setting up the current problem (XIN, WIN, METRIC and STEP).

SUBROUTINE SSFLAG - Sets flags in cells adjacent to a solid surface boundary condition.

SUBROUTINE WIN - Reads in the five flow variables from the restart file.

SUBROUTINE XIN - Reads in coordinate triples from either the grid file or the restart file.

FLOW SOLUTION ROUTINES

SUBROUTINE EULER - The main work routine; sums fluxes; integrates in time; applies enthalpy damping; calculates residuals; calls artificial dissipation routine (FILTER); calls IRS routine (PSMOO)

SUBROUTINE FILTER - Calculates the artificial dissipation needed for stability.

SUBROUTINE PSMOO - Implicitly smoothes residuals to accelerate convergence to a steady-state.

SUBROUTINE STEP - Calculates ratio of cell time step to cell volume for time independent solutions. Calculates the global minimum time step for time accurate flows.

BOUNDARY CONDITION ROUTINES

SUBROUTINE BC - Calls the three boundary condition routines, RIEMANN, SOLID and POLE.

SUBROUTINE POLE - Properly loads data into ghost cells in regions where boundary cells have zero face area. The

loading ensures proper application of the artificial dissipation at a polar singularity.

SUBROUTINE RIEMANN - Enforces far field, sink and source boundary conditions via an isenthalpic Riemann Invariant procedure.

SUBROUTINE SOLID - Calculates the pressure on solid surfaces and symmetry planes.

I/O ROUTINES

SUBROUTINE BLKIN - Inputs grid, flow, metric and time step variables from scratch files.

SUBROUTINE BLKOUT - Outputs grid, flow, metric and time step variables to scratch files.

SUBROUTINE IN - Main intersection variables input routine.

SUBROUTINE OUT - Main intersection variables output routine.

SUBROUTINE SWAP - Manages data when a block abuts itself.

CONVERGENCE MEASUREMENT ROUTINES

SUBROUTINE CONVRG - Calculates maximum and RMS mass residuals and number of supersonic cells. Calls FORCMOM.

SUBROUTINE FORCMOM - Calculates integrated forces and moments.

MISCELLANEOUS OUTPUT ROUTINES

SUBROUTINE DIAGNOS - Holds plot file; generates data for plotting.

SUBROUTINE PLOT - Makes the actual plots of Cp, Mach No., etc.

SUBROUTINE SAVESOL - Writes grid and flow variables to new restart file.

REFERENCES

1. Jameson, A., Schmidt, W. and Turkel, E., "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes," AIAA-81-1259, June 1981.

2. Jameson, A. and Baker, T. J., "Solution of the Euler Equations for Complex Configurations," AIAA-83-1292-CP, July 1983.

3. Moitra, A., Turkel, E. and Kumar, A., "Application of a Runge-Kutta Scheme for High-Speed Inviscid Internal Flows," AIAA-86-0104, January 1986.

4. Strang, W. Z., "Aspects of an Out-Of-Core Solver," to be published.

5. Strang, W. Z., "QBERT: A Grid Evaluation Code," AFWAL-TM-88-193, July 1988.

6. Strang, W. Z., "MERCHEK User's Manual," AFWAL-TM-88-216, July 1988.

7. Buning, P. G., "PLOT3D User Documentation," NAS Documentation Number ZD-9408-00-N00.

8. Cray X-MP and Cray-1 Computer Systems, Library Reference Manual, SR-0014, Copyright 1984, Cray Research, Inc.

9. Kirkland, C., "AQIO Users Guide," Cray Research, Inc.

10. AQIO User's Guide, SN-0247, Cray Research, Inc.